UNIVERSITY OF LIVERPOOL

PHONOCARDIOGRAM CLASSIFICATION USING MOTIF DISCOVERY

Thesis submitted in accordance with the requirements of the University of Liverpool for the degree of Doctor in Philosophy by

Hajar Alhijailan

January 2021

Abstract

The research work presented in this thesis is directed at diagnosing heart diseases using Phonocardiograms (plots of heartbeat sound recordings). The central motivation was to provide point of care diagnosis using machine learning software shipped with digital stethoscopes; what might then be referred to as *intelligent digital stethoscopes*. Thus the aim was to classify Phonocardiograms (PCGs) using time series analysis techniques, more specifically motif-based time series analysis. The main challenge was the size of the PCG time series to be considered.

The main contributions of the thesis are four approaches: (i) the MK Benchmark to PCG classification, (ii) PCG_{seg} Classification, (iii) SGR-FMD and (iv) CE-FCS approaches. The MK Benchmark approach investigated, for the first time, the application of motif discovery to PCG data. The fundamental rationale of this approach was to provide a vehicle with which to compare the alternative approaches presented in the thesis. The PCG_{seg} Classification approach provided a novel bespoke segmentation technique, based on "shapes", which served to significantly reduce the processing time (by a factor of more than eleven) compared with the Benchmark approach. The fundamental rationale underpinning the SGR-FMD approach was to prune the time series data by removing sub-sequences that were unlikely to be representative of any class in order to reduce the complexity of the motif discovery process. In more detail, the rationale was to remove the "silent gaps" from the PCG data. The SGR-FMD approach also featured a novel technique of clustering. The runtime was improved by a factor of more than 278 compared with the PCG_{seg} Classification approach. The CE-FCS approach rationale was to generate meaningful motifs while at the same time reducing the number of computations. This was applied to the PCG recordings by extracting the heart cycles that represented potential motifs and by considering the statistical distribution of these motifs. This approach produced the best results of the four approaches proposed in this thesis; its accuracy was the highest recorded and the application runtime was the least.

The evaluation data set used was a canine PCG data set obtained from the School of Veterinary Science, Small Animal Teaching Hospital, the University of Liverpool, who collaborated on the research. This data set featured four classes, and was labelled by domain experts.

Acknowledgements

I would like to thank my first supervisor Prof. Frans Coenen for his support, encouragement and cooperation throughout my PhD journey. I really appreciate his positive energy that he was broadcasting, especially during the difficult days. I feel very honoured to have this experience with him. My sincere thanks are also to my advisors, Dr. Yannis Goulermas and Dr. Keith Dures for their valuable feedback and constructive assessment of my research.

I am thankful to Prof. Jo Dukes-McEwan and Mr. Xavi Navarro-Cubas, veterinary cardiologists and academics at the Veterinary School's Small Animal Teaching Hospital, the University of Liverpool, who provided unlimited support by welcoming us during our visits to their clinic, providing valuable recommendations and recording the audio files used in this research.

My sincere and profound gratitude goes to my whole family who have been supported me so that I can follow my dreams.

Last but not the least, I would like to gratefully acknowledge the financial support of my beloved homeland so that my dream can come true.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Overview

Heart disease is the leading cause of mortality worldwide [47, 163]; globally one in three individuals die from heart diseases, one death is recorded every four minutes in the UK [47]. Heart diseases is one of the most expensive medical conditions to treat; the annual cost in England alone has been estimated at £15.8 billion by Public Health England in 2019 [47]. The cost includes, but is not limited to: diagnostic tests, monitoring, hospitalisation, surgery, specialist visits and medicines. Both patients with heart conditions who have not been diagnosed until their conditions have become complicated (false negatives) and normal subjects who are being sent to specialists or inpatient care (false positives) contribute to this cost. The later, false positives, are very common because, understandably, practitioners tend to err on the side of caution. For instance in mitral valve disease, the overwhelming majority (80%) of cases diagnosed by primary care physicians and referred to cardiologists are for healthy subjects [164].

One way of minimising faulty diagnosis and therefore the total cost, not to mention other negative effects, such as disease worsening and weak productivity (in the case of false negatives), and frustration and fear in patients (in the case of false positives), is to provide Artificial Intelligence (AI) support for the diagnosis process. The advantages that AI can provide for health care have been well documented. The need for AI technology in healthcare was identified in the UK Life Sciences Industrial Strategy report [24], has been named as the principal driver for personalised healthcare [105] and as providing a potential solution to the UK National Health Service's predicted "budget gap" of £30 billion by 2021 [5]. Examples of where AI can benefit healthcare and wellbeing were identified in the UK Hall and Pesenti report [56], where one specific example was support for data-driven diagnosis.

The process of heart condition diagnosis usually commences with a physician or General Practitioner (GP) using a traditional stethoscope to monitor a subject's heartbeat. In the case of cardiac disease, arrhythmia and/or abnormal sounds, such as murmurs and clicks, are indicators that something is not right. Not all cases are seen by a specialist, some are sent home (preclinical cases) and others require referral to a cardiologist for further tests. The usage of stethoscopes for diagnosis requires skill on behalf of the practitioner and consequently may entail human error because of the time needed to acquire the necessary skills and experience [40, 53]. Consequently, it can be the case that subjects are sent home with an undetected heart condition that is not diagnosed until the condition develops further and physical symptoms appear and/or further complications arise [53]. Although relying only on a stethoscope for diagnosis has been shown to lead to erroneous diagnosis, it remains an important tool in: cardiology [177], the evaluation of congenital cardiac defects [168] and primary home health care [168]. Another advantage is that the traditional stethoscope is inexpensive, hand-sized and noninvasive [40].

A recent innovation is the replacement of the traditional stethoscope with digital/ electronic stethoscopes. An example of an electronic (digital) stethoscope is shown in Figure 1.1. This is the *eKuore* Vet electronic veterinary stethoscope, which connects via wireless technology to an application compatible with Android and iOS. The heartbeat is thus available in a digital form, referred to as a Phonocardiogram or PCG, opening the way to the application of AI techniques, especially machine learning techniques, to enhance diagnosis as noted above. It is also interesting to note that the information contained in PCGs is more than can be distinguished by the human ear or by visual inspection of the signal trace.

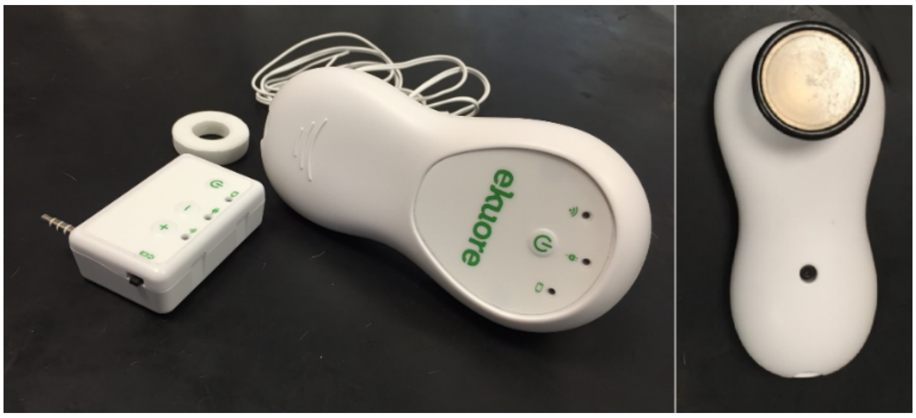


Figure 1.1: Electronic *eKuore* veterinary stethoscope

Essentially, a PCG is a time series comprised of time stamps and amplitude values. The application of time series analysis, the process of extracting knowledge from time stamped data, therefore seems appropriate. The usual application for time series analysis is the construction of a classification model for labelling (classifying) previously unseen time series [12, 50, 69, 112, 171, 178]. Many techniques have been proposed to analyse time series data, examples include: Fuzzy Logic [156], Artificial Neural Networks [44], Hidden Markov Models [52], Genetic Algorithms [90], Support Vector Machines [70], Self-Organising Maps [14] and Bayesian approaches [131].

The application of time series analysis techniques to PCG data is thus an obvious step, examples can be found in [8, 19, 51, 102, 129, 137, 142, 166, 175, 187]. A range of mechanisms have been proposed, frequently involving some kind of sequence segmentation to isolate individual heartbeats; a process facilitated by coupling the PCG data with Electrocardiogram (ECG) data [30]. An ECG is similar to a PCG except that an ECG describes electrical activity whereas a PCG describes sound activity; both are illustrated in Figure 1.2. However, given an outpatient clinic scenario, a GP surgery scenario or a home visit scenario, ECG data is unlikely to be available. ECG machines, which record the electrical signals produced by the heart, are expensive and subject to availability. The output also requires visual inspection by trained specialist physicians whose availability is also limited; this is especially the case in developing countries and/or rural areas [53]. Ideally, we would like point of care diagnosis using machine learning software shipped with digital stethoscopes; what might then be referred to as intelligent digital stethoscopes. This is the central motivation for the work presented in this thesis.

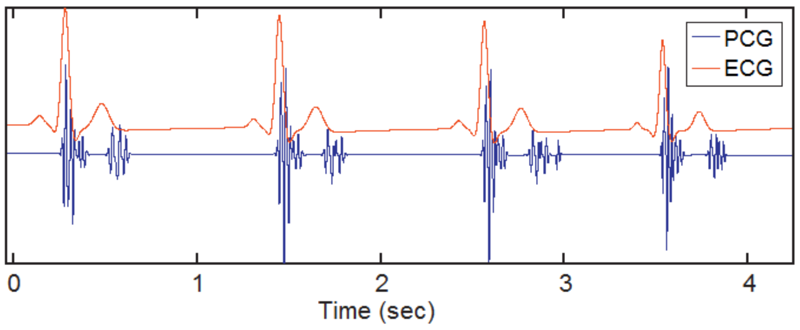


Figure 1.2: PCG and ECG signal traces, time versus amplitude

A further motivation for the work presented in this thesis is that the PCG time series of interest are typically very large. The average number of points (length) per second in a PCG time series is in the thousands. PCG time series are thus typically too large to be considered in their entirety given the standard computing power available in doctors' surgeries; for example as a single feature vector. One way of addressing this issue is by identifying *motifs* within the time series [36, 39, 50, 150, 165, 171, 176]. A motif in this context is some sub-sequence of points occurring within a time series which is deemed to be representative of the underlying class-label associated with the time series [85, 92]. A representative motif can be defined in various ways; one approach is to consider frequency of occurrence [9], another approach is to consider pattern similarity between candidate motifs [112]. The discovered motifs can then be used to label (classify) previously unseen time series [12, 50, 112, 171]. However, finding motifs that are good representatives of class-labels is computationally challenging, especially with larg time series (as in the case of PCG data), mainly because of the large number of candidate motifs that need to be considered. To the best knowledge of the author, at time of writing, the concept of motifs had not been considered in the context of PCG data.

1.2 Research Question

From the outset, the motivation for the work presented in this thesis is the need for AI support to provide point of care diagnosis by applying time series analysis techniques to PCG data, without the support of accompanying ECG data, in a manner that is both efficient and effective. The fundamental idea is to use the concept of motifs to build a classification model. The challenge is how this can best be achieved. The research question to be answered is thus:

What are the most appropriate mechanisms that can be used to identify indicative patterns (motifs) in previously unseen PCG data in a manner that is both efficient and effective for the purpose of PCG data classification?

The resolution of this research question entailed the resolution of a number of related subsidiary research questions:

1. What is the most appropriate mechanism to identify "motifs" that are indicative of some conditions within the PCG time series of interest?
2. With regard to (1), how can this mechanism be applied in a tractable manner, given the challenge of processing large PCG data time series?
3. How can the identified motifs be best used to generate a PCG classification model?
4. How can the generated PCG classification model best be used to provide point of care, near real time, diagnosis?

1.3 Research Methodology

The adopted research methodology to provide an answer to the above research question and subsidiary questions was to start with a benchmark algorithm taken from the literature and produce a first pass at generating a solution. To this end, the MK algorithm (proposed by Abdullah **M**ueen and Eamonn **K**eogh) was selected [112]. To train and test the model, a canine PCG data set was obtained from the School of Veterinary Science who were collaborating on the project. This data set featured four classes. A number of publicly available PCG data sets were available for download [25, 46, 119, 174], however, as will be discussed later in this thesis, their usage within the context of the specific research question that this thesis sought to address was found to be less than straightforward.

The adopted evaluation strategy was to conduct the analysis by considering the standard classification model evaluation metrics. These included: accuracy (acc), precision (prec), recall (rec) and F-score (f-s), which have been used to evaluate multi-class classification models [155] and are calculated using a confusion matrix comprised of the number of True Positives (TP), True Negatives (TN), False Positives (FP) and False Negatives

(FN). For a multi-class model applied to a set of classes C of size $|C|$, the above metrics were calculated as follows:

$$acc = \frac{1}{|c|} \times \sum_{i=1}^{|c|} \frac{TP_i + TN_i}{TP_i + TN_i + FP_i + FN_i} \tag{1.1}$$

$$prec = \frac{1}{|c|} \times \sum_{i=1}^{|c|} \frac{TP_i}{TP_i + FP_i} \tag{1.2}$$

$$rec = \frac{1}{|c|} \times \sum_{i=1}^{|c|} \frac{TP_i}{TP_i + FN_i} \tag{1.3}$$

$$f\text{–}s = \frac{2}{|c|} \times \sum_{i=1}^{|c|} \frac{prec_i \times rec_i}{prec_i + rec_i} \tag{1.4}$$

Note that the above metrics are commonly utilised in the classification literature. Moreover, the runtime of the proposed solution was the main subject of interest. In addition, Five-fold Cross-Validation (FCV) was used whereby the given (labelled) data was stratified and divided into five equally sized parts. Using FCV, the classifier was trained and tested five times, each time using a different four-fifth of the data and tested on the remaining fifth. The results, presented later in this thesis, are thus the average of the 5 folds.

Once the initial benchmark approach had been developed and the analysis completed, work could commence on improving on this benchmark in terms of effectiveness and efficiency. A sequence of approaches was envisioned each featuring an improvement on the previous approach. Of course, any proposed new approach needed to provide some advantage over the benchmark approach in order to be worthwhile. The intention was to explore a number of ideas. These included techniques for pre-processing the data to reduce its overall size with a focus on segmentation pre-processing. Some further detail concerning the various techniques considered and tested is provided in the following section where the contribution of the work presented in this thesis is discussed.

1.4 Contributions

Through the adoption of the foregoing research methodology, three novel problem-specific approaches were proposed in addition to a benchmark approach. These represent the four main contributions of the thesis. For each approach, a variety of techniques were adopted to enhance the quality criteria (efficiency and/or effectiveness). A brief review of these contributions is given below.

- **Contribution 1, The MK Benchmark PCG Classification Approach:** The principle idea of the Benchmark approach was to provide a vehicle with which to compare the three alternative approaches proposed later in the thesis. The aim was to analyse the operation of this approach in the context of the Phonocardiogram application domain. A number of techniques were used in this approach:

  1. A tractable, not approximate, algorithm (the MK algorithm [112]) for Phonocardiogram classification.
  2. The "Sample" representation method. The typical method of representing PCG signals is as Amplitude series; a novel method is to represent it as Sample series. The novel method showed a reduction in the data size by approximately half.
  3. The similarity-based representation strategy for identifying motifs that were "representative" of a class.
  4. An exclusion technique to exclude "trivial" comparisons from consideration during the motif discovery process as an in-process technique.
  5. Process termination using the concept of "early abandonment" when conducting similarity comparison for classification purposes, hence providing efficiency gains.
  6. A new classification model, Smallest Average Classification (SAC), developed by the author; a variation of Nearest Neighbour Classifier. It takes into consideration the similarity between a new motif to be labelled and all motifs for each class in the "motif bank".

  The evaluation of this approach indicated that some form of pre-processing of the data was an essential requirement if the runtime issue was to be addressed. This is discussed further in Chapter 5.

- **Contribution 2, The PCG$_{\mathbf{seg}}$ Classification Approach:** This approach was designed to address the issue of the overall size of the PCG time series to be processed. The techniques incorporated into this approach, to improve upon the benchmark approach in terms of efficiency and effectiveness, were as follows:

  1. The "Sample" representation method, which reduced the overall data size.
  2. A novel bespoke, two layer, time series segmentation technique, the PCG$_{seg}$ technique, to transform the data representation. It was proposed to further reduce the data size; the number of segments was much less than the number of points.
  3. Two novel mechanisms to measure similarity when using the proposed two-level hierarchical segmentation: Strict and Tolerant Similarity Measurement.
  4. A process termination technique for use with the proposed two-level segmentation to allow early abandonment to avoid the need of going down to the bottom

level of the segmentation if the top level did not satisfy prescribed similarity conditions.

5. The similarity-based representation strategy, the exclusion technique for "trivial" matches, "early abandonment" and the SAC classification model as proposed with respect to the Benchmark approach.

The PCG_{seg} approach improved the runtime (by a factor of more than eleven) without adversely affecting the resulting accuracy. This is discussed further in Chapter 6.

- **Contribution 3, The SGR-FMD Approach:** The Silent Gap Removal - Frequent Motif Detection approach was designed to reduce the complexity of the motif discovery process by pre-processing the data in preparation for motif discovery. The fundamental idea was to prune the time series by removing sub-sequences that were unlikely to be representative of any class-label. This approach used the following techniques to enhance the quality criteria:

1. A frequency-based representation strategy for identifying "representative" motifs of a class.
2. In addition to the exclusion technique for "trivial" matches, an exclusion technique, to enhance motif selection, used as a pre-processing technique, the PCG_{sgr} technique. It excluded some parts of the data on the grounds that they were irrelevant to the analysis task at hand; namely, excluding the silent gaps that appear in the PCG signals.
3. Two levels of representativeness were considered: local and global. The first was widely used in the literature, where a motif is selected at the single time series level. Using the local technique, time series that might be the result of background noise could be selected as being representative of a class. Given an unseen record to be classified which features the same background noise, the record could be labelled based on that noise. The local technique therefore has some disadvantages. Consequently, by using global representativeness the effect of unwanted noise could be avoided.
4. A pruning technique used the novel "zero-motif" concept; a base motif with which all sub-sequences in the data can be compared. This technique retained the potential discriminative sub-sequences which led to reduce the data size, hence the required runtime. This technique also served to eliminate the randomness used in selecting potential motifs and to reduce the number of computations.
5. A bespoke pruning technique involving clustering, which further pruned infrequent sub-sequences from the data. This technique was also used for randomness elimination and computation reduction.

6. A selection technique for computation reduction based on “best of the best” idea, where two selection thresholds, *max* and *k*, were applied to a set of potential motifs obtained from previously pruned data.
7. Conflict resolution methods to address the issue whereby unlabelled time series to be classified using the SGR-FMD approach ended up with a number of class-labels, where one was required. The three methods considered to select the most appropriate class-label were: Shortest Distance, Shortest Total Distances and Highest Votes.
8. The process termination “early abandonment” concept also used in the previous two proposed approaches.

The SGR-FMD approach served to significantly reduce the processing time (by a factor of more than 278) compared to the previous approach, the PCG_{seg} approach. This is then the third contribution of the thesis and is discussed in further detail in Chapter 7.

- **Contribution 4, The CE-FCS Approach:** The Cycle Extraction - Frequent Cycle Selection approach was designed to address both motif quality and the required processing time by generating meaningful motifs and, at the same time, reducing the number of computations. This was achieved using the following techniques:

1. A “motif generator” technique, the PCG_{ce} technique, for extracting meaningful motifs, which was designed to isolate heartbeat cycles. A fundamental feature in this generator was that it considered all data points in time series in the generation process.
2. The frequency-based representation strategy.
3. The trivial-match exclusion technique.
4. The process termination using “early abandonment” concept.
5. The local and global representativeness idea.
6. The “zero-motif” pruning technique.
7. A second pruning technique that considered the statistical distribution of data to exclude irrelevant sub-sequences, which made the motif discovery process tractable.
8. The “best of the best” selection technique.
9. The conflict resolution methods from the previous approach.

The CE-FCS approach was found to be the best of the four presented approaches; its performance was compared to all the approaches proposed in this thesis as discussed in detail in Chapter 8. This is then the fourth, and last, contribution of the thesis.

1.5 Publications

A number of academic papers have resulted from the work presented in this thesis as follows:

- Alhijailan H., Coenen F., Dukes-McEwan J., Thiyagalingam J. (2018). *Segmenting Sound Waves to Support Phonocardiogram Analysis: The PCGseg Approach.* In: Xin Geng, Byeong-Ho Kang (eds) PRICAI 2018: Trends in Artificial Intelligence, Proceedings of PRICAI 2018, Springer Lecture Notes in Computer Science, Vol 11013, pp. 100-112, Springer. Paper presented the second of the above contributions and formed the foundation of the material discussed in Chapter 6.

- Alhijailan H., Coenen F. (2019). *Effective Frequent Motif Discovery for Long Time Series Classification: A Study Using Phonocardiogram.* In: Ana Fred, Joaquim Filipe (eds) IC3K 2018: Knowledge Discovery and Information Retrieval, Proceedings of IC3K 2019, Vol 1, pp. 266-273, ScitePress. Paper detailing the third approach considered in this thesis and contributing to the content of Chapter 7.

- Alhijailan H., Coenen F. (2019). *Effective Frequent Motif Discovery in Phonocardiograms.* Submitted to the Knowledge and Information Systems journal. A journal paper detailing the third approach found in Chapter 7.

- Alhijailan H., Coenen F. (2019). *Motif Discovery in Long Time Series: Classifying Phonocardiograms.* In: Max Bramer, Miltos Petridis (eds) SGAI-AI 2019: Artificial Intelligence, Proceedings of SGAI 2019, Springer Lecture Notes in Artificial Intelligence, Vol 11927, pp. 198-212, Springer. Paper describing the fourth of the above listed contributions, that presented in details in Chapter 8.

- Alhijailan H., Coenen F. (2020). *Fast Time Series Classification Using a New Concept for Motif in Phonocardiograms.* In preparation, to be submitted to the Expert Systems with Applications journal. Journal paper extending the previous paper by providing further detail and founded on the content of Chapter 8.

1.6 Structure of Thesis

The rest of this thesis is structured as follows. Chapter 2 gives a review of the relevant previous work followed by a description of the data sets used in this thesis in Chapter 3. A formalism for the work presented in this thesis is then given in Chapter 4. The benchmark approach, to which all other approaches proposed in this thesis were compared, is presented in Chapter 5. Chapters 6, 7 and 8 then detail the proposed PCG time series classification approaches, the main contribution of the thesis. The thesis is completed with some concluding remarks in Chapter 9.

1.7 Summary

This opening chapter has provided the motivation for the work presented in this thesis and the underpinning research question. In summary, the main objective of this thesis is to investigate mechanisms whereby PCG signals can be processed so as to generate classification models that can then be used to label previously unseen data. This opening chapter has also detailed the adopted research methodology, the various contributions of the thesis and the structure of the thesis. The chapter was completed with a brief review of a number of papers that have resulted (or will result) from the work described. In the following chapter, a literature review, relevant to the work considered in this thesis, is presented.

Chapter 2

Literature Review

2.1 Introduction

As noted in the previous chapter, the work conducted in this thesis intersects across two established research areas: time series analysis and PCG classification. This chapter presents the necessary background concerning these research areas with respect to the work presented later in this thesis. The chapter is divided into five sections, including this introductory section. Section 2.2 considers time series analysis and the concept of motifs. Section 2.3 considers time series pre-processing techniques designed to reduce the time complexity of the analysis, with a particular focus on the segmenting of time series because it was used significantly with respect to the work presented later in this thesis. In Section 2.4, a review of the PCG data sets used for evaluation purposes is presented. The chapter is concluded with a summary in Section 2.5.

2.2 Time Series Analysis

As the name implies, time series analysis is concerned with the processing of time series data. More specifically, it is directed at the extraction of knowledge from temporally referenced data, for example using supervised and unsupervised learning techniques, usually in the context of a single variable [12, 57, 60, 112, 183], as in the case of the work presented in this thesis, although research on multi-variate time series analysis has been undertaken [31, 37, 65, 140, 170, 172, 183, 189]. Time series data can be argued to be a specialised form of point series data [27]; in both cases, the distinguishing feature of such data is that it comprises an ordered sequence of values. In the case of time series data, the ordering is defined by a sequence of time stamps; in the case of point series data, the ordering is defined by a sequence of indices (which might be interpreted as time stamps). Time series analysis methods can be divided into two types: time- and frequency-domain methods. In time-domain methods, it is the ordering of sequences of point values, which is also referred to as "temporal precision", that is important. Meanwhile, frequency-domain methods consider the number of times values appear in a series, which is also referred to as "spectral precision" [22, 59]. It is the time-domain methods which are of interest with respect to the work presented in this thesis, because of the repetitive nature of the data used in this thesis (PCGs); we are interested in identifying repetitive patterns (motifs) that recur over time within PCGs [29].

The common motivations for time series analysis, in the context of data mining, pattern recognition and machine learning, include: (i) time series clustering and classification [13, 71, 73, 82, 102], (ii) the querying of time series databases [82, 152], (iii) anomaly (outlier) detection [20, 173], (iv) time series forecasting [104], (v) time series indexing [32, 78, 191] and (vi) time series modelling of the domain from which the time series are drawn [52, 162]. For the work presented in this thesis, the intention was that the desired analysis would be conducted using supervised learning techniques, which require labelled time series training data. Using these learning techniques, a model that is able to predict the labels of previously unseen time series data could be produced. The process of learning and using such a model is called classification [146] and is illustrated in Figure 2.1. The colour coding in the figure is used to distinguish: (i) the training of the classifier (blue), (ii) the testing of the classifier (black) and (iii) the usage of the classifier (red).

The remainder of this section is divided into three sub-sections. Sub-section 2.2.1 presents a review of time series supervised learning techniques (time series classification), whilst Sub-section 2.2.2 considers the literature concerning PCG classification, the application domain of interest with respect to this thesis. Sub-section 2.2.3 then presents the idea of motif discovery, the particular time series analysis supervised learning technique that dominates the work presented in this thesis motivated partly by the observation that the technique has not previously been applied to PCG data.

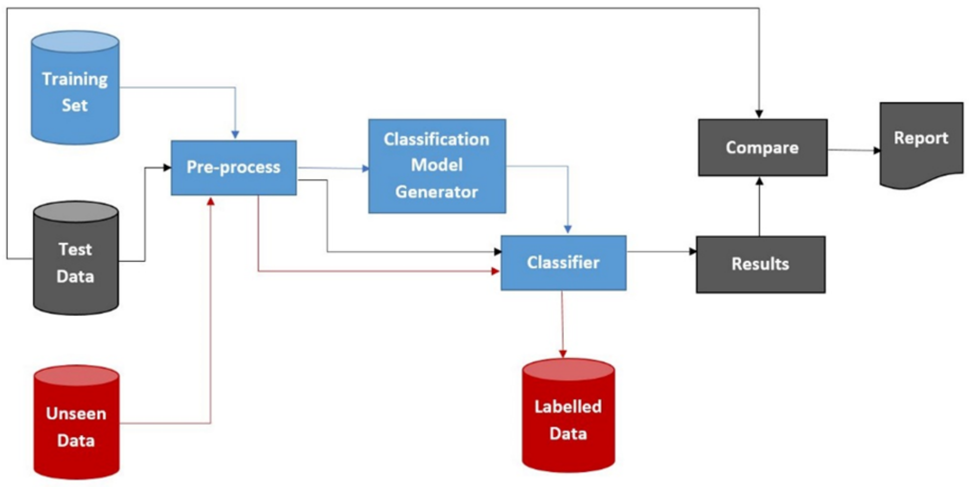


Figure 2.1: Time series classification (supervised learning) model generation and usage; blue: training phase, black: testing phase, red: usage phase

2.2.1 Time Series Classification

Classification is the process of generating a model, using pre-labelled training data, which can be used to label (classify) previously unseen examples. Early forms of classifiers, before the age of Machine Learning (ML), were designed based on rules sometimes expressed in the form of a decision tree. Given any substantial data collection, such as a PCG time series data collection, it becomes difficult to extracting such rules in an effective manner. Since the advent of ML, the classification rule learning process can be automated [178]. However, the domain of ML includes much more sophisticated techniques for learning classification models that do not rely on the concept of rules.

The most common classification approaches used with time series data can be categorised in a variety of different ways. One approach classifies time data series according to three different concepts: similarity, probability and boundary. In similarity-based classification, the classifier estimates a class-label of an unlabelled time series according to its similarity with an existing collection of labelled time series (a training set) [112]. Nearest Neighbours Classification (NNC) is the most commonly used approach in this category. This is therefore also the approach used with respect to the work presented later in this thesis, because of its simplicity and effectiveness [97]. Using the second category of approach, the classifier outputs the predicted class-label of an unlabelled time series in terms of a probabilistic distribution over the set of classes [8]. From the literature, Bayes decision rules, density estimation and maximum likelihood are examples of probability-based classifiers. The third category of approach involves the construct of decision boundaries in the training data whereby the classifier can decide to which class-label an unlabelled time series belongs [187]. This is usually done by optimising certain error criterion; for example, by minimising the empirical error as in Neural Networks (NNs), or by maximising the empirical margin surrounding a decision boundary as in the case of Support Vector Machine (SVM) classification.

2.2.2 PCG Classification

Previous work on PCG classification has mostly been directed at diagnostic purposes [19, 51, 71, 89, 102, 116, 128, 132, 142, 143, 144, 166, 173, 187], although some work has been done on authentication using PCGs [8, 129, 198]. A variety of time series classification methods, such as Neural Networks (NNs) [19, 51, 89, 102, 116, 132, 142, 144], Support Vector Machine (SVM) [142, 166, 187], Nearest Neighbours Classifier (NNC) [130, 135], hidden markov models [147, 181], decision tree [145], random forest [116] and adaBoost classification [132], have been applied to PCG data.

The challenge of applying classification techniques to PCG data is the size of the individual time series, which can run to millions of points. Applying the above techniques to entire PCG time series is therefore not a practical solution given the capabilities of the current average computer. The solution is to pre-process the PCG data, prior to classification model generation and subsequent usage, so as to reduce the size and/or computation requirements. The above referenced research, directed at the automated classification of PCGs, used some form of pre-processing. Time series pre-processing, and by extension PCG pre-processing, is therefore discussed in further detail in Sub-section 2.3.3.

Another mechanism for avoiding the need to process entire time series, especially where the time series are lengthy, is to use sub-sequences within the data known as *discriminative patterns* [10, 52, 77, 112]. To the best knowledge of the author, there has been no work directed at the PCG diagnostic problem that has used discriminative patterns. One type of pattern, which is of particular interest with respect to the work presented in this thesis, is known as a *motif* [36, 55, 68]. More discussion concerning motifs is presented in the next sub-section, Sub-section 2.2.3.

2.2.3 Motif Discovery

A motif is a reoccurring sub-sequence in a time series. There are various ways of defining and identifying motifs, a formal definition is presented in Chapter 4. A simple working definition is that a motif is a time series sub-sequence that has at least one non-trivial match with another sub-sequence in a given time series according to some predefined similarity threshold [107, 169]. Note that a "trivial match" is where a sub-sequence overlaps with the candidate motif, hence the two are bound to feature similarity. To measure how well two sub-sequences match, a distance function is required. Euclidean Distance (ED) is widely used in the literature [21, 36, 73, 77, 112, 198] with some evidence suggesting its competitiveness with, or superiority to, other more complex measures [42]. The time complexity of computing the ED between two time series sub-sequences of length g is $O(g)$.

The simplest and most straightforward motif discovery algorithm is a Brute Force algorithm; comparing every candidate motif (sub-sequence) against every other candidate motif. To identify a motif, there are different comparison criteria for a potential sub-sequence to be considered a motif. Two strategies were adopted to select a motif from a single time series with respect to the work presented in this thesis:

1. Select the candidate motif that has the highest similarity score compared to all other candidate motifs in the time series.

2. Select the candidate motif that occurs most frequently compared to all other candidate motifs (assuming we only want to identify one motif in a given time series).

For both strategies, a threshold is usually set to define similarity (using exact matching for frequency counting often results in a frequency score of one). The simplest brute force algorithm with respect to the first strategy is to compute the similarity values between all sub-sequences (using a nested loop) and maintaining a best similarity so far. The simplest brute force algorithm with respect to the second strategy is to compute the similarity values between all sub-sequences (again using a nested loop) and generating a count for each sub-sequence which is incremented by one every time a similar alternative sub-sequence is found (similarity decided according to some threshold).

Both brute force algorithms have a complexity of $O(n^2g)$ assuming ED was used for similarity comparisons, where n is the number of time series sub-sequences and g is the length of each sub-sequence. Whatever the case, for any significant time series, both algorithms require significant resource [81, 112]. A number of more efficient, but approximate, motif discovery algorithms have therefore been proposed [94, 95, 96], while a tractable exact algorithm remains a research challenge [18, 112]. The research focus of this thesis is, in part, the latter. Schemes aimed at reducing the above complexity [36, 39, 50, 112, 150, 165, 171, 176] remain problematic in that the performance of applications using the discovered motifs tends to be adversely affected, typically because of the nature of the various proposed heuristics used to limit the time complexity. However, there are two generic techniques that can be usefully employed to increase the efficiency of the motif discovery process.

- *Non-trivial matches:* The simplest technique is to restrict the number of comparisons by excluding trivial matches [36]. Recall, that trivial matches exist where two sub-sequences to be compared overlap; in other words, the sub-sequences to be compared share data points and therefore can be expected to feature some similarity. This technique is widely used in many proposed motif discovery algorithms [36, 112] and is adopted with respect to the work presented later in this thesis.

- *Early abandonment:* Another technique for reducing the complexity of the motif discovery process is to adopt the concept of "early abandonment" whereby a similarity comparison is stopped when the dissimilarity between two sub-sequences being compared reaches some threshold when it can safely be assumed that the two time series are not similar. The well-known MK motif discovery algorithm [112] features early abandonment, as do the motif discovery algorithms presented in this thesis. The threshold can be user-defined; alternatively, as in the case of [112], it can be derived by conducting r rounds of comparisons so as to establish a lower bound for the sought after similarity.

Two significant issues that this thesis seeks to address in the context of motif discovery applied to PCG data, are: (i) the time to generate/identify meaningful motifs and (ii) the accuracy of the consequent application results. The first problem is exacerbated by the size of PCG time series. The second is concerned with the quality of the identifiable motifs; they need to be good differentiators of class. The two issues are inter-linked in that, if motif generation can be made more efficient, additional resource will be available to select "better" motifs given a fixed time constraint. A suggested solution to the first issue, and consequently a possible solution to the second, is to pre-process the time series so as to reduce their size in such a way that salient features are preserved [80]. Some time series pre-processing techniques to support motif discovery are discussed in the following section, Section 2.3.

2.3 Pre-processing

This section provides an overview of existing work directed at reducing the time complexity of time series analysis algorithms, and by extension motif discovery algorithms, in order to enhance their efficiency and effectiveness. Time series pre-processing can be viewed as the transformation of an original time series into another series in such a way that important information is kept whilst, at the same time, the time series analysis becomes more tractable. The section starts, Sub-section 2.3.1 by considering time series pre-processing in general. One particular time series pre-processing technique, that of segmentation, because of its significance with respect to this thesis, is singled out and discussed in particular detail in Sub-section 2.3.2. Sub-section 2.3.3 considers PCG time series pre-processing in particular, because of the relevance to the work presented in this thesis. The section is concluded, Sub-section 2.3.4, with a review of existing literature on segmentation as applied to PCG data, again because of its relevance with respect to the work presented in this thesis.

2.3.1 Time Series Pre-processing

This sub-section provides a brief overview of time series pre-processing as applied in the general time series analysis context, as opposed to the specific PCG context considered later in this section. Time series pre-processing can be conducted either in a generic manner or in a specific manner using knowledge specific to the application domain under consideration. We refer to techniques that fall into the first category as *Comprehensive techniques* and those that fall into the second as *Exclusive techniques*. Each is discussed in further detail in this sub-section.

Popular comprehensive time series pre-processing techniques include:

- *Dimension reduction* [88]: The transformation of high-dimensional time series into low-dimensional time series; the number of dimensions could be reduced to 1 [63, 165]. Clearly this technique is only applicable to multivariate time series.
- *Denoising* [109]: The removal of the noise from the data if identifiable and known to exist.
- *Segmentation* [12, 15, 66, 77, 80, 126, 193, 194]: Dividing the time series into blocks representing the components of the time series. This is one of the main techniques considered with respect to the pre-processing of PCG data presented in this thesis, and is therefore discussed in further detail in Sub-section 2.3.2.
- *Down-sampling* [21, 149]: Collapsing or combining sequences of points into single points in some regular manner.
- *Filtering* [64, 109]: Filtering the time series to reveal significant features; this is referred to as smoothing or low-pass filtering when directed at identifying long term trends, and high-pass filtering or de-trending to identify isolated events.

- *Decimation* [41, 75, 148]: Literally retaining every tenth point, but more generally a combination of smoothing and down-sampling of the data.

It should be noted that, whatever the case, any adopted comprehensive time series pre-processing technique should be selected carefully so as not to adversely affect any future data analysis. It has been suggested that the selection of an appropriate technique is dependent on the domain from which the data is obtained [111]. The application of any of the above comprehensive techniques will clearly lead to a reduced processing requirement, for example a lower number of comparisons.

Exclusive techniques, as opposed to comprehensive techniques, aim to reduce the overall size of a given time series by excluding some parts of the time series on the grounds that they are not relevant to the time series analysis task at hand. To achieve this reduction, some form of domain knowledge is required. For example, in the case of motif discovery, knowledge of "the distance between" or "the position of" relevant candidate motifs can be fruitfully used to exclude some comparisons from consideration in a given time series. The simplest exclusive time series pre-processing technique is the removal of irrelevant sub-sequences because they are known not to be relevant. For instance, when working with DNA sequences [192], it is possible to exclude some sub-sequences because they are known in advance to be irrelevant.

Although most of the techniques in both categories (comprehensive and exclusive techniques) could be theoretically applied to the data used in this thesis (PCG recordings), there remain some concerns. Techniques like down-sampling, filtering and decimation are by their nature random, therefore accuracy is not guaranteed. In addition, there may still be a substantial computational overhead. Moreover, dimension reduction and de-noising are not applicable to PCGs, as considered in this thesis, because the used PCG data is one-dimensional in nature and the noise content is not known in advance. Given specific PCG related domain knowledge, this can clearly be used in the context of exclusive pre-processing techniques directed at PCG time series data in particular. Existing work directed at the pre-processing of PCG data is therefore considered in more detail in Sub-section 2.3.3.

2.3.2 Time Series Segmentation

Segmentation, as noted in Sub-section 2.3.1 above, is one of the standard pre-processing techniques applied to time series data. The main idea is to coarsen the data by dividing it up into "chunks" whereby each chunk is associated with some specific characteristics. In other words, segmentation is the process of dividing a whole entity into its constituent parts or distinct elements, the term is frequently used in the context of image analysis [23, 117]. Work has also been conducted using audio creation and editing software such as Adobe Audition, however this is largely a manual process. Segmentation can reduce the number of sub-sequences and hence the number of comparisons and is therefore of significant benefit with respect to motif generation. Segmentation has been used extensively with respect to the work presented in this thesis; specific contributions are presented in Chapters 6 and 8.

Techniques for achieving effective and efficient segmentation remain an area of current research. In the context of time series in general, a number of mechanisms have been used to achieve segmentation, these include: (i) Fourier Transforms [10, 79, 187], (ii) Wavelets [19, 126, 137, 151], (iii) Symbolic Mappings (SM) [15, 62] and (iv) Piecewise Linear Representation (PLR) [66, 72, 80, 82, 125, 152, 180, 186, 195, 196]. Of these, PLR is the most common. PLR is also of relevance with respect to the work presented in Chapter 6.

The fundamental idea of PLR is to translate a given time series P into a model $\bar{P}$ which comprises a number of "best fitting" straight lines (segments). The PLR technique has been used in a variety of contexts including: (i) time series clustering and classification [82] and (ii) the querying of time series databases [103]. Examples of the latter includes fuzzy querying [152] and weighted querying [82]. The application of PLR can be specified in different ways: (i) a specific number of line segments, (ii) the similarity of line segments to the entire given point series (according to how well the overall linearisation matches the given point series) and (iii) the similarity of individual line segments with respect to the individual time series sub-sequence they represent.

However, regardless of the segmentation approach used, each can be implemented using one of three basic mechanisms as follows [80]:

1. **Sliding Window:** Using the sliding window mechanism, the time series is processed using a "window" that is slid along the time series from the start point to the end point. Usually the window size, ω, is dynamic. On commencement, ω is set to a default value and the first window is grown until the segment is no longer representative of the point series sub-sequence in the window, at which point the segment is captured, and the window moved so that it starts at the end of the current segment with ω again set to the default value, and so on.

2. **Top Down:** Using this mechanism, the point series is repeatedly sub-divided into a certain number of parts, each representing a segment, and each of which is tested according to some parameter(s). For each segment that is sufficiently similar to the point series it represents, the segment will be stored. The remaining segments will then be sub-divided further. And so on.

3. **Bottom Up:** This mechanism is the opposite of the Top Down mechanism. Using the Bottom Up mechanism, the given point series is divided into the largest possible number of segments. Segments are then recursively merged. Each merged segment that does not meet some criteria will be merged further. Those segments that do meet the required criteria are stored for later use.

The first approach is faster than the other two, because it operates in a sequential manner whereby each segment is identified in turn. However, the point (time) series is not considered in its entirety, decisions are made without any deep exploration of the time series as in the case of the latter two approaches; this in turn may affect the quality of the segmentation. Both Top Down and Bottom Up mechanisms give good results, however they tend to be impractical for large time series as they require a scan of the entire time series. One of the methods proposed in this thesis uses the Sliding Window mechanism, however the segmentation is not related to similarity with the underlying point series, but instead with how accurately the segments represent the "shape" of the underlying point series sub-sequence; not quite the same thing. This is discussed in much further detail in Chapter 6.

In the context of line segmentation, and for completeness, it is worth noting that there are two main techniques for representing segments as straight lines: (i) Linear Interpolation and (ii) Linear Regression. The first is a straightforward and very fast technique, which simply draws a straight line between the beginning and the end of each segment aligning the endpoints of consecutive segments. In contrast, the linear regression technique has a high computational complexity, compared with linear interpolation, because it determines the best fitting line for all points in each segment. In this manner, a sequence of disjoint lines are produced. The latter is thus argued to provide a better quality segmentation, however, as noted in [16], the same line segment can result from very different sequences of points. This is evidenced, presented in [16], using the *Anscombe's quartet* reproduced here in Figure 2.2, which demonstrated that the same line segment could be produced using linear regression applied to four very different point series.

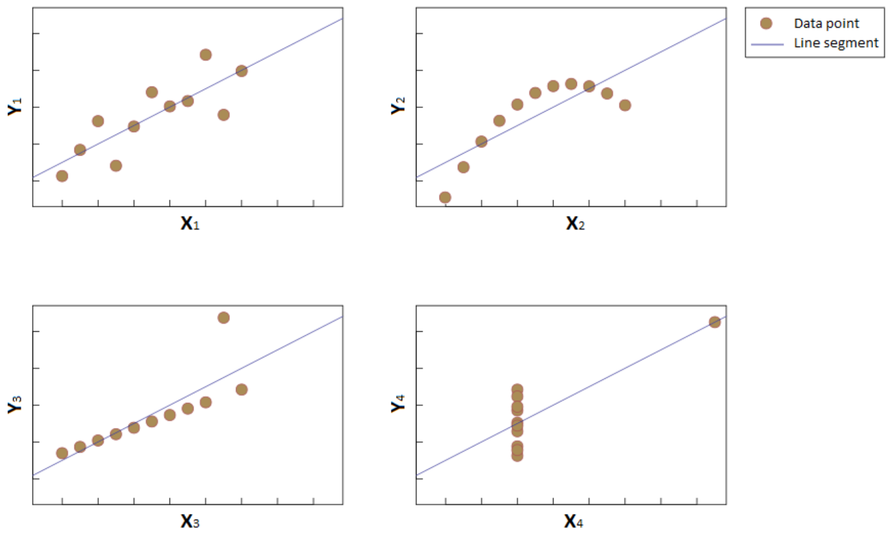


Figure 2.2: The "Anscombe's quartet" illustrating the disadvantage of Linear Regression line fitting whereby four very different point series are represented by the same line segment [16]

For the time series used in this thesis, PCG signals, the segmentation was achieved in a similar manner to PLR, using a sliding window approach and the linear regression technique, but with some differences; the segments are a collection of a limited number of shapes whereas their sub-segments represent trends.

It should also be noted that in the field of sound file segmentation, there is work in the context of speech recognition [26, 49, 84, 118, 123] where the focus is on segmenting human speech in the form of WAVE files, the same data format as used with respect to the PCG data files considered in this thesis. The idea is to identify "phonetic" segments representing words, syllables or letter boundaries (in different languages). Most of this work focuses on segmenting short sentences, the aim being to achieve language understanding regardless of accent and other factors, whilst the focus of PCG file segmentation, as conceived of in this thesis, is to identify motifs that are representative of class. Novel mechanisms for segmenting PCG WAVE files, as generated by hand-held stethoscopes, are one of the contributions of this thesis.

2.3.3 PCG Pre-processing

From the literature, most existing work on automated PCG data analysis is concerned with computer-aided support for PCG analysts rather than fully automated analysis. In particular, much reference is made to the use of graphical tools for this purpose [98, 114]. There is little reported research on automated PCG pre-processing for analysis (even if only in a coarse manner). Current work on pre-processing PCG recordings has focused mainly on three techniques: (i) dimensionality reduction, (ii) signal denoising and (iii) heartbeat segmentation. The first two are discussed in further detail below. Segmentation is of particular relevance with respect to the work presented in this thesis and is therefore discussed more extensively in the following sub-section, Sub-section 2.3.4. A further pre-processing technique, of particular relevance to PCG data, is "silent gap removal". This has been adopted with respect to the work presented in this thesis and is therefore considered in further detail at the end of this section.

The primary purpose of dimensionality reduction, as noted earlier, is to improve learning performance [167]; the fundamental idea is to retain only distinctive features of the data while at the same time reducing the overall size of the data. Dimensionality reduction in the PCG domain is typically conducted in two stages: feature extraction and feature selection. The former can be conducted in: (i) the time-domain (for example, using linear features [142]), (ii) the frequency-domain (for example, using short-time Fourier transforms [129]) or (iii) a combination of the two [8, 102, 166, 175]. The initial identified set of features is processed in the second stage where the idea is to retain only the most relevant features, in other words, a subset of the identified feature set. How relevance is defined in this context depends on what we wish to do with the data, but typically we wish to avoid features that are irrelevant and/or redundant. From the literature, frequently cited feature selection techniques that have been used in the context of PCG data include: Principal Component Analysis (PCA) [102, 142], Genetic Algorithms (GA) [175] and Generalised Discriminant Analysis (GDA) [142]. The nine most common feature domains used for PCG data regarding feature selection are: (i) time interval, (ii) state Amplitude, (iii) energy, (iv) entropy, (v) cepstrum, (vi) cyclostationarity, (vii) frequency spectrum of states, (viii) frequency spectrum of records and (ix) high-order statistics [8, 71, 102, 129, 142, 166, 175, 187]. However, the PCGs in the work presented in this thesis are interpreted as one-dimensional time series, so dimension reduction is not applicable in this case.

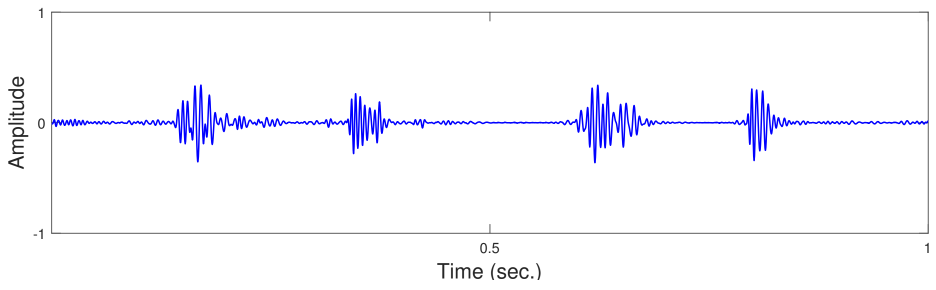
Notch filtering, adaptive filtering, averaging and wavelet decomposition are all common denoising mechanisms that have been adopted for signal denoising in the context of PCG data [198]. However, there is no work directed at the removal of specific categories of noise from PCG signals, which can be corrupted with surrounding noise or, in a controlled environment, with instrumentation noise, respiratory noise, fetal breath sounds and so on [6], and/or internal body organs sounds [197]. Most of the research directed at PCG signal denoising requires a separate reference signal [91, 120, 134, 139]. However, this assumes such a reference signal is available; not the case with respect to the hand-held electronic stethoscope scenario considered in this thesis. An alternative frequently cited method, when no reference signal is available, is to use some form of Wavelet Transform (WT) [43, 51, 188, 121]. The main drawback of using WTs is the need for users to define a set of parameters which in turn will affect the quality of the use of the WT, for example: (i) the need to pre-specify a "mother wavelet", (ii) the need to choose an applicable decomposition level, and (iii) the need to select a thresholding method [54, 106, 143, 198]. Other work directed at PCG denoising has tended to be directed at specific categories of noise, such as high-energy noises. Generally speaking, the quality of the PCG output signal, when using any of the above forms of denoising, is adversely affected by the application of the denoising [100, 128, 173]. This somewhat defeats the objective of the denoising. The question, however, is the significance of this effect, which in turn is application dependent.

In Sub-section 2.3.1, it was noted that given domain knowledge, it is possible to prune (pre-process) time series by removing parts that are known not to be relevant; the techniques used for this purpose were referred to as exclusive techniques. In the case of PCG data, a particular feature is that of "silent gaps" between heartbeats. Intuitively, such silent gaps can be excluded because they do not hold relevant information with respect to motif discovery and classification. Silent gap removal is an exclusive pre-processing technique frequently used with respect to applications that are founded on audio data. Silent gap removal was first proposed and adopted in the context of applications directed at voice recordings [76]; more specifically Voice Activity Detection (VAD) and speech recognition [74, 83, 138, 154, 190]. The main motivation for removing silent gaps in VAD and speech recognition was that these sub-sequences were not likely to carry any information [190]; by removing the silent gaps, the problem domain becomes more tractable [74, 190]. Of course silent gap identification also allows for the isolation of individual words, syllables and sentences [138, 154]. To the best knowledge of the author, there is no work on silent gap removal in the context of audio-recorded heartbeat (PCG) data, although it has a clear role to play with respect to reducing the overall size of the sound signals. The idea of silent gap removal was thus incorporated into the third PCG classification approach presented in this thesis, the Silent Gap Removal - Frequent Motif Detection (SGR-FMD) approach presented in Chapter 7.

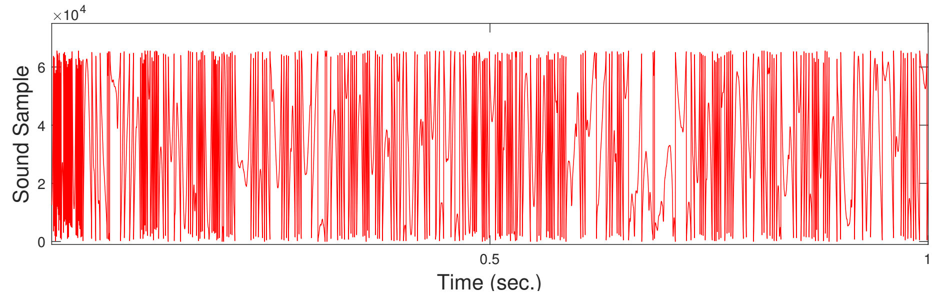
2.3.4 PCG Segmentation

In the previous sub-section, where PCG data pre-processing was considered, it was noted that PCG segmentation is of particular relevance with respect to the work presented in this thesis. Segmentation in general was discussed in Sub-section 2.3.2. In the context of PCG time series, once the time series has been segmented, it is possible to identify significant features and patterns across the segmented sub-sequences which are indicative of some conditions [51, 187]. Alternatively, it is possible to produce a spectrum, a visual representation of the frequencies in a sound signal.

The manner in which PCG data can be segmented depends on the nature of the signal representation. The usual way of representing PCG signals is as an Amplitude signal (Figure 2.3(a)); an alternative mechanism, and that considered in this thesis because it fits well with the nature of the data produced by hand-held electronic stethoscopes, is to represent the PCG data as a Sample signal (Figure 2.3(b)). Both are considered in this Sub-section. Amplitude-PCG signal segmentation, which has been studied extensively in the existing literature, is reviewed in Sub-section 2.3.4.1 below. Sample-PCG segmentation is then considered in Sub-section 2.3.4.2, this is only briefly discussed because, to the best knowledge of the author, it has received no attention in the literature.



(a) An Amplitude-PCG point series



(b) A Sample-PCG point series

Figure 2.3: A point series with two different representations, Amplitude and Sample

2.3.4.1 PCG as Amplitudes

Segmentation approaches reported in the literature that have interpreted PCG signals as Amplitude time series have typically been used as a mechanism for isolating patterns in time series, given knowledge of the nature of the time series. PCGs feature cardiac cycles, each comprised of four basic components: (i) the first cardiac sound (S1), (ii) systole, (iii) the second cardiac sound (S2) and (iv) diastole. The segmentation of PCGs to identify these cardiac components is a common research topic in the field of Signal Processing [35, 58, 113, 122, 139, 197]. Segmentation is applied to isolate cycles and their components.

Cardiac cycle PCG segmentation is usually achieved with respect to a reference signal, either an ECG signal recorded at the same time and/or a Carotid Pulse (CP) [91, 120, 134, 139]. In the case of PCG signals, collected using electronic stethoscopes in veterinary clinics, the application focus of the work presented in this thesis, no such reference signal was available. This is also applicable in the majority of cases where only an electronic stethoscope is available to conduct diagnoses. In such cases, the components of a PCG signal can still be extracted by processing the signal. Some reported research directed at the segmentation of PCGs without a reference signal exists [11, 43, 51, 61, 93, 108, 188, 121]. Some of these [43, 93, 121], concentrate on finding the first and second cardiac cycle components ($S1$ and $S2$) in a "waveshape". Others, such as [51], concentrate on extracting all four cardiac cycle components and/or heartbeat events. The major limitations of this previous work are that: (i) empirical methods are typically used whose performance is affected by noise [108] or murmurs [11, 61, 188], and (ii) the focus of the majority of reported cardiac cycle component detection techniques is directed at normal cardiac activity; whereas this thesis aims at classifying both regular and irregular cardiac activities.

In the absence of reference signals, segmenting Amplitude-PCG signals is typically achieved according to the "energy" of the signal and one or more energy thresholds [35, 58]. The well-known Shannon Energy is frequently used [35, 58, 110] as it maintains time series features. There are situations where not all of the features are needed, as in the case of the work presented in this thesis, in which case alternative energy methods can be used as long as the required salient features are preserved.

It is argued that extracting the cardiac components from PCGs using empirical thresholds is inappropriate because of the varying Amplitudes recorded [197]. This is due to difference between subjects in: (i) the thickness of the chest wall [197], (ii) subject age [177], (iii) subject mood [177] and (iv) further subjective factors [177]. Alternative methods have therefore been proposed. The simplest methods still use thresholds but dynamically computed [35, 58], whilst others use much more sophisticated methods, such as methods based on acoustic characteristics [34] or Gaussian regression of "smoothed simplicity profiles" [133]. The work presented in this thesis adopts a dynamic threshold approach because of its straightforward deployment.

2.3.4.2 PCG as Samples

An innovative way of representing PCG time series is as a Sample signal. This was the format used with respect to the work presented in this thesis, because this is the format produced by the hand-held electronic stethoscopes of interest with respect to this thesis. There are many ways of representing audio data, but the format used in this thesis was the Waveform Audio File or WAVE format (extension `.wav`). In this format, each Sample contains one or more Amplitudes depending on the audio file specification, more specifically depending on the used bit-depth. The common bit-depths (bit/sample) are 8-bit, 16-bit and 32-bit. Among them, 16-bit is the most widely used audio format in PCG recordings, which means each Sample in the audio stream consists of two bytes, leading to more preciseness, resulting in high fidelity audio. Given that each Amplitude is saved in a byte, each Sample holds two Amplitude values. The innovative way used in this thesis was to represent each point in the point series by a Sample instead of an Amplitude. In other words, to have a series of Samples instead of a series of Amplitudes. This way of representing the time series leads to a significant reduction in the length by a half.

There is no work, to the best knowledge of the author, directed at the segmentation of the Sample-PCG representation, because the main goal of most of the PCG research is to investigate Amplitude features. Another reason for the lack of research on Sample-PCGs might be because of the difficulty in obtaining this kind of data. The aim in this thesis was to discover motifs in the PCG data that can be used for diagnostic purposes. Visual inspection of Sample time series (Figure 2.3(b)) indicates that some reoccurring patterns exist, therefore it is clear that segmentation is applicable to Sample time series. However, the nature of Sample-PCG time series is such that neither PLR, nor any of the other techniques presented above, are appropriate. In the case of the PLR and SM techniques, line fitting and symbols are not sufficiently descriptive for the purpose of PCG classification. In the case of Fourier transforms and wavelets, these work in the frequency-domain while motif discovery is better conducted using the time-domain. More detail concerning the PCG data used in the literature for diagnostic purposes is presented in the next section, Section 2.4.

2.4 PCG Data

As noted earlier in the introductory chapter, there are a number of challenges associated with heartbeat manual analysis; a process known as auscultation. These can be itemised as follows:

- Auscultation requires substantial experience [40].
- The auscultation device used affects the outcome sound [124].
- Sound data from the same subject will vary according to mood [177].

- Heartbeat patterns change with age [177].
- Thickness of the chest wall [197].
- Noise from external sounds included in the recording, such as organs noise, background noise and friction between the stethoscope chest-piece and the subject's skin (or hair or fur in the case of animal subjects) [40, 197].

A further issue with recorded PCG audio data is data privacy. In the medical field, as in other fields, data privacy is a significant issue; real patient PCG audio data is therefore difficult to obtain. This issue is the main reason for the lack publicly available labelled PCG data resources. Other factors are the needs for domain experts (to label the data), and time resource from both patients and physicians. A further technical challenge is that the data requires a large memory resource since the sound files are large.

Despite the existence of the above obstacles, the large percentage of erroneous diagnoses of heart diseases, which contributes enormously to the high rate of mortality and morbidity [47, 163, 164], has been the motivation for most of the research in the field of PCG analysis. The broad objective of the existing works, including the work presented in this thesis, was to investigate mechanisms whereby PCG signal data can be analysed in terms of some kind of supervised learning mechanism and consequently classified according to some set of predefined labels.

Supervised learning requires labelled data. In response to this need, some organisations have made the decision to release data for use with respect to technical competitions; one in 2012 [25] and another in 2016 [174]. This "online" data availability has resulted in a "shift" in the field of PCG diagnosis with an increase in the number of studies [71, 102, 166, 173, 187] despite the existence of some limitations, such as modifications to the data [25, 174] and artificial additions [4] (more details are provided in the next chapter, Chapter 3). In addition, it should be noted that some PCG data released online was not intended for diseases diagnosis purposes, but for authentication purposes [158] or denoising purposes [100]. The most common way of obtaining data to support research on PCG analysis is to collect it privately, as in the case of the work presented in this thesis. This private data might be obtained from university hospitals [108, 129, 198] or from hospitals cooperating with researchers [4, 136].

2.5 Summary

This chapter has provided a review of time series analysis and classification techniques with a particular focus on the pre-processing of such data. The chapter has included an overview of existing work directed at motif discovery; and a review of existing work concerned with reducing the search time for time series motif discovery in order to enhance the analysis efficiency. Specific detail has been provided with regard to existing research concerning the segmentation of time (point) series, especially PCG time series because of the relevance with respect to the work presented in this thesis. The chapter was concluded with a discussion of the availability of PCG data which is considered in further detail in the following chapter, Chapter 3, where a description of the data used in this thesis is presented.

It is worth noting that in this chapter some pre-processing techniques used in this thesis have been mentioned. These are summarised below:

- **Segmentation:** A common pre-processing technique is time series segmentation. The basic process is to divide the time series into blocks. From the literature, a variety of algorithms have been proposed to obtain a good high-level segmentation of time series data. A proposed PCG segmentation method is presented in Chapter 6.
- **Pruning:** Another pre-processing technique is the pruning of time series. The main idea is to omit some sub-sequences in a given time series that are not deemed to be relevant. There are a variety of mechanisms by which a time series can be pruned. A novel proposed PCG pruning mechanism is detailed in Chapter 7.
- **Cycle Extraction:** The last pre-processing technique considered in this chapter and of particular relevance to the work presented in this thesis is cycle extraction. This is discussed further in Chapter 8; cycle extraction is aimed at reducing the number of computations needed later in an analysis process. This is not a commonly used technique for Motif Discovery in time series; however, with knowledge of the application domain, this technique is proposed in Chapter 8.

Chapter 3

Evaluation Data Acquisition

3.1 Introduction

A fundamental precursor to any form of machine learning is the acquisition of data sets from which some desired model can be learned and evaluated. For many machine learning applications, well-documented benchmark training/test data sets are available. One of the most frequently used repositories is the UCI Machine Learning Repository [3], which is available online and contains, at time of writing, 481 data sets. Among them, 94 are time series data sets. However, there are no Phonocardiogram (PCG) data sets held in the UCI repository. There are some PCG data sets available for download from various locations that have been used in the context of research directed at PCG analysis that has been reported in the literature. This short chapter commences, Section 3.2, with a review of the application domain used as focus for the work presented in this thesis, building on discussion from earlier chapters. The chapter then goes on, Section 3.3, to consider the existing PCG data sets that have been referenced in the literature and that are available for download. Next, Section 3.4 presents a comprehensive overview of the bespoke canine PCG data set specifically collected to support the work presented in this thesis. The chapter is concluded, Section 3.5, with a summary of the contents of this chapter.

3.2 Application Domain

Before considering the relevant data sets, a review is provided in this section of the application domain so as to provide the context for the remainder of the chapter. The traditional approach to analysing the sound of the heart's vibrations is by auscultation [101], the process of listening to the internal sounds of a living body. This is usually conducted using a stethoscope. An electronic or digital stethoscope (a Phonocardiograph) produces a two-dimensional, time versus Amplitude, plot of the recorded sound which is commonly referred to as a Phonocardiogram (PCG) as shown early in Figure 1.2.

There are a number of conditions that can be identified from such PCGs. Some of these can be argued to be relatively common, for instance: mitral regurgitation, patent ductus arteriosus, and pulmonic and aortic stenosis. Each can be further categorised as being mild, moderate or severe. Considering only common conditions will entail consideration of some thirty class -abels; note that multi-class classification is significantly more challenging than binary classification, especially when considering a large number of classes [7].

Using PCG data, and appropriate computer software, various signal processing techniques can be applied [45] so that diagnoses can be automatically generated. However, usage of PCGs has not gained wide acceptance; partly because of perceived flaws in the quality of the recording technology (doubts over whether the signal trace is a faithful rendition of heart vibration) and also partly because of a lack of quantitative techniques for reliable analysis. For example, the *eKuore* Phonocardiograph (electronic stethoscope), pictured earlier in Figure 1.1, is reported to produce good sound but is said to lose some quality when played through speakers. However, it can be assumed that, if a faithful rendition is not produced, the signal trace will at least be an indicator of diseases/no-disease. Manufacturers of electronic stethoscopes, such as the *eKuore*, are of course keen to increase the uptake of these devises, which in turn requires clear evidence of the benefits. Note that manufacturers of electronic stethoscopes typically bundle their product with software to visualise PCG signals, typically with functions to highlight elements of the signal; however currently no sophisticated, machine learning based, automated analysis is provided.

One of the main challenges in the domain of machine learning for PCG analysis is the lack of pre-labelled data with which to train the necessary models. The main reason for this is the data confidentiality requirements associated with PCG data; PCGs are a form of personal data whose usage requires patient consent (assuming PCGs obtained from humans) and ethical approval. Many patients and their families, for a range of reasons, are reluctant to allow their data to be released to non-medical bodies. At the organisation level, some private health-providers and commercial companies are also reluctant to share their data for reasons of commercial confidentiality and competition. Where data has been obtained, following an ethical approval process, there is typically a condition that it can only be used by the named researchers, or named institution, itemised in the ethical approval application; thus preventing publication of the data. Consequently, example PCG data is not substantially available for use by the wider research community. However, there is a number of exceptions as noted in the following section.

3.3 Public Databases

To the best knowledge of the author, there are a few public heart sound data sets available online: (i) the Cardiac Auscultation of Heart Murmurs (eGeneralMedical) database, (ii) the Michigan Heart Sound and murmur (UMHS) database, (iii) the sixth community-based Signal Separation Evaluation Campaign (SiSEC2016) biomedical signals (BIO2016) database, (iv) the Heart Sounds Catania 2011 (HSCT11) database (v) the Pattern Analysis, Statistical Modelling and Computational Learning (PASCAL) database and (vi) the Shiraz PhysioNet/Computing in Cardiology (PhysioNet/CinC) database. These have all been used with respect to work reported in the literature [71, 100, 102, 158, 166, 173, 187, 198]. Some further detail concerning these six databases is given below:

1. The eGeneralMedical database [46] is provided by eGeneral Medical Inc. It is not a freely available open data set, but requires payment. It contains 64 recordings for 49 heart conditions, which means most of the conditions have only one associated record and thus it is unsuited for training a classifier.

2. The UMHS database [119] is available from the University of Michigan. It comprises 23 recordings, each for a separate heart condition, therefore it is also unsuitable for training a classifier.

3. The SiSEC2016-BIO2016 database [4] was provided as a challenge sponsored by Native Instrument. The competition was organised in conjunction with the 13th International conference on Latent Variable Analysis and Independent Component Analysis (LVA/ICA 2017) held in Grenoble in France. The challenge was to denoise 16 unlabelled recordings from 3 healthy participants. The noise was artificially generated and the recordings were bandpass-filtered (15 - 300 Hz). All the recordings are unlabelled, which means the database is also inappropriate for training a classifier.

4. The HSCT11 database [157] is provided by Andrea Spadaccini and Francesco Beritelli and collected by the University of Catania in Italy. It was collected from 206 participants, two recordings for each person, with a total of 412 recordings. The participants comprised 157 males and 49 females. The providers used it in a biometric system where the data set was divided into two parts: 206 recordings (one recording/person) for the training phase and the other 206 recordings for the test phase. This database is suitable for authentication not diagnosis applications.

5. The PASCAL database [25] was released as a digital challenge associated with a workshop colocated with the 15th international conference on Artificial Intelligence and Statistics (AISTATS 2012) held in La Palma in the Canary Islands. The challenge was sponsored by PASCAL (now Kaggle) hence the name. The challenge was to segment the data into two heart sounds and then to classify some 247 unlabelled recordings using 585 labelled recordings associated with 5 heart conditions, including normal. All the recordings underwent a low-pass filter resulting in a frequency range of below 195 Hz.

6. The PhysioNet/CinC database [174] was provided by Shiraz University in Iran for a challenge that took place in 2016. The challenge was to classify the recordings into three categories: normal (12% of the recordings), abnormal (77%) or too noisy to be classified (11%). The data set consisted of 4,430 recordings which were resampled to 2,000 Hz. The data was divided into 6 data sets, each was collected using a different stethoscope. Moreover, the location of the recording from the subjects was not unified; this means it could be any one of the four usual locations for recording.

The first two databases, given the number of records per disease, are obviously not appropriate for diagnostic (classification) tasks in machine learning. They were in fact intended to teach medical students auscultation [99]. The third database is unlabelled, modified and small so cannot be used for diagnosis, it was released for the task of denoising [100]. The forth database is used widely in the field of heart sounds biometry [158, 198], but its labelling system is not suitable for the field of diseases diagnosis. The last two contain relatively large numbers of recordings suitable for training/testing a classification model. However, both were modified (resampled), with the twin consequences that many of the heart sound components, used with respect to the work presented in this thesis, were removed and that the "purity" of the data was compromised. Although the signal frequency range was shortened, in most of the examples, the noise is louder than the heartbeat. A further disadvantage with respect to PhysioNet/CinC, and again with respect to the work presented in this thesis, is that only two classes are featured (excluding the "too noisy" class) and hence the data set is only suitable in the context of binary classification.

3.4 The Canine PCG Data Set

Given the disadvantages associated with the publicly available data sets considered in the foregoing section, and the more general difficulties of collecting PCG data from humans, as discussed in Section 3.2 above, a canine PCG data set was used. Indeed, the potential availability of such data was part of the motivation for the research presented in this thesis. This data set was curated by the School of Veterinary Science at the University of Liverpool, who collaborated with the author with respect to the work presented in this thesis. The data was collected using the *eKuore* electronic stethoscope shown previously in Figure 1.1. Using *eKuore*, the data was stored in Waveform Audio File (WAVE) format (file extension `.wav`). This is a commonly used format for raw, uncompressed, audio data that can easily be translated into other formats (the MATLAB computing environment provides functions to do this). The PCG recordings were interpreted as time (point) series. Two representations were used with respect to the work presented in this thesis: (i) Amplitude (Figure 2.3(a)) or (ii) Sample (Figure 2.3(b)). The data set compromised 59 canine PCG recordings collected from 18 dogs; some with Mitral Valve disease and the rest with normal heart function.

All the collected Mitral Valve disease recordings were labelled, by domain experts, according to the stages of the disease. Mitral Valve disease can be divided into four basic stages: A, B (includes further subdivision: B_1, B_2), C and D according to the European College of Veterinary Internal Medicine (ECVIM) classification [17, 115]. The collected data set did not feature any Stage A or D recordings. Each recording (point series) in the collected database therefore had a class-label associated with it selected from the class-attribute set $\{B_1, B_2, C, Control\}$ where the first three class-attributes were stages of Mitral Valve disease and the last class-attribute represented recordings that did not feature any disease (used for control purposes). A summary of the data used is given in

Table 3.1. The collated canine PCG data set was the data set used for evaluation purposes with respect to the work presented in this thesis.

Table 3.1: Summary of evaluation data

Class	**No. of Patients**	**No. of PCGs**	**Size of Each PCG** (in KB)				
Stage B_1	2	2	1774	1340			
		5	1825	1825	1825	1849	1825
Stage B_2	7	2	2954	1942			
		3	2106	1608	1782		
		3	2094	1344	3806		
		4	1825	1825	1825	1861	
		4	804	1416	1770	1732	
		4	898	676	1268	2848	
		5	2642	2260	888	1068	2610
Stage C	5	1	1825				
		2	1810	2804			
		3	1825	1825	1849		
		4	1448	1360	1918	2242	
		5	1794	1890	830	1326	1162
Control	4	2	3074	2538			
		3	2004	2288	2232		
		3	2686	1682	2112		
		4	2542	1788	1238	3868	
Total	18	59	174345				

3.5 Summary

This chapter has provided an overview of the PCG application domain, publicly available PCG data, and the Canine PCG data set curated with respect to the work presented in this thesis. It was noted that, for a variety of reasons, both commercial and regulatory, there are few publicly available PCG data sets that are available for academic research purposes. Six specific PCG data sets, frequently referenced in the literature, were described: eGeneralMedical, UMHS, SiSEC2016-BIO2016, HSCT11, PASCAL and PhysioNet/CinC. However, as noted in the text, none of these could be considered as being suitable for providing a generic benchmark data set for the purpose of comparing PCG classification model generation mechanisms. Instead, a Canine PCG data set, curated especially for the work presented in this thesis, was used with respect to the evaluation presented in this thesis. The Canine PCG data set was therefore described in detail in this chapter.

Chapter 4

Formalism

4.1 Introduction

This chapter presents the formalism for the work presented in this thesis. The thesis uses some new concepts, such as the "zero-motif" concept; the formalism presented in this chapter is therefore a necessary prerequisite for understanding the remaining chapters in the thesis. The definitions and notation used are presented in Section 4.2. The chapter is concluded, Section 4.3, with a summary of the contents of this chapter.

4.2 Definitions

Some key terms used in this thesis are defined bellow. The relationship between these definitions is illustrated in Figure 4.1. A list of the significant symbols used is given in Table 4.1.

Definition 1, Time Series: A time (point) series P is a sequence of data values $\{p_1, p_2, \dots\}$ associated with a class-label c_i taken from a set of classes $C = \{c_1, c_2, \dots\}$. In the case of the training and test data, used for classification model construction and validation, this label is known. In the case of previously unseen data, this is what the classifier is intended to predict. In this thesis, a time series is a heartbeat signal (a PCG) which consists of a number of cardiac cycles. A collection of labelled point series is then given by $T = \{\langle P_1, c_1\rangle, \langle P_2, c_2\rangle, \dots\}$ where each P_i is a point series and $c_i \in C$.

Definition 2, Point: A point p in a time (point) series P is a numerical data value. In this thesis, a time series is an interpretation of a PCG recording. Two interpretations were used separately. The first was as a series of Amplitude values, where p is a signed 8-bit number. The second interpretation was as a series of Sample values, where p is an unsigned 16-bit number.

Definition 3, Segmented Time Series: A segmented time (point) series $\overline{P}$ consists of a sequence of segments $\{S_1, S_2, \dots\}$ where each segment S_i is a sub-sequence of a

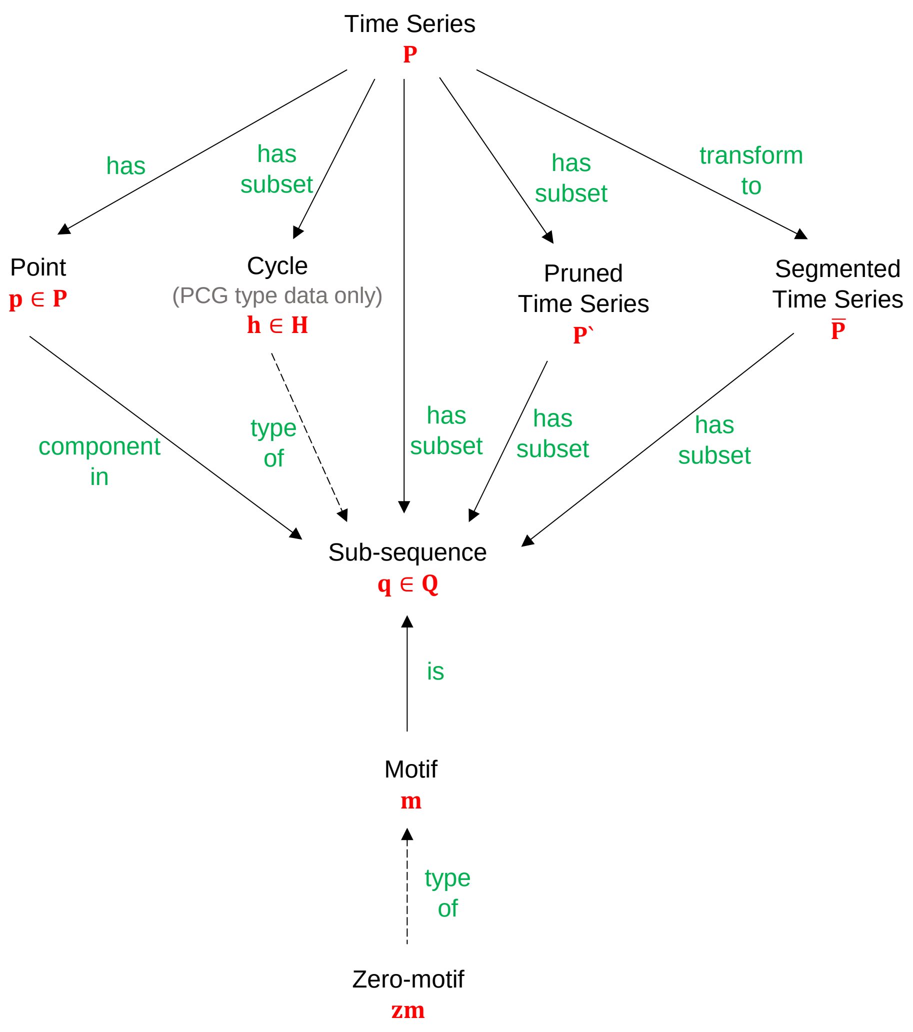


Figure 4.1: The relationship between some used concepts

given point series P. In this thesis, segmentation is conducted in a two-tier hierarchical manner. Each segment is defined in terms of a tuple of the form $\langle S_p, S_C \rangle$, where S_p is the parent segment and S_C is a set of constituent sub-segments $\{S_{c_1}, S_{c_2}, \dots\}$. S_p, in turn, is defined in terms of a tuple of the form:

$$\langle shape, type, length \rangle$$

where: (i) *shape* is the nature of the point series defined by the segment, {slant, vertical, dome, flat}; (ii) *type* is direction of the shape, either "up" or "down"; and (iii) *length* is the number of points represented by the segment. More specifically, the length of a segment is the difference between the start and end index values. Thus, a shape comprised of two points will have length 1 and so on.

Each constituent sub-segment $S_{c_i} \in S_C$ is represented by a second tuple of the form:

$$\langle type, length, depth \rangle$$

where: (i) *type* is the direction of the point series defined by the sub-segment {up, down, flat}, (ii) *length* is the length of the sub-segment (calculated as described above), and (iii) *depth* is the difference between the maximum and minimum point values represented by the sub-segment in question. Note that only a two-tier hierarchy was used because this was all that was required to permit an appropriate level of comparison without adversely affecting the operation of the system, although from a technical perspective additional levels might be possible.

Definition 4, Pruned Time Series: A pruned time (point) series P' is a sequence of data values $\{p_1, p_2, \dots\}$ such that $|P'| \leq |P|$. In other words, it is a time series P with one or more points removed, for example, those points that are considered to represent noise.

Definition 5, Time Series Sub-sequence: A time series sub-sequence q is any consecutive set of points in $\mathbf{P}$, where $\mathbf{P}$ could be P, P' or $\overline{P}$. The set Q is the set of all possible sub-sequences, of length ω, in $\mathbf{P}$; $Q = \{q_1, q_2, \dots, q_{|\mathbf{P}|-\omega+1}\}$. The generation of Q is computationally expensive because, given any reasonably sized time series, there will be $|\mathbf{P}| - \omega + 1$ possible sub-sequences. This can be limited by generating only a pre-specified number of sub-sequences defined by a variable *max*; thus $Q = \{q_1, q_2, \dots, q_{max}\}$.

Definition 6, Cycle: A cycle is a special form of sub-sequence (Definition 5) that represents a heartbeat cycle h which is a subset of a time series P that represents a PCG. A cycle is a single complete heartbeat comprised of two principal components: the first and second sounds; in the medical field these two components are referred to as $S1$ and $S2$ respectively. A collection of labelled cycles is then given by $H = \{\langle h_1, c_1\rangle, \langle h_2, c_2\rangle, \dots\}$ where each h_i is a heartbeat cycle and $c_i \in C$. A set H is therefore like a set Q, but with the inclusion of class-labels.

Definition 7, Candidate Frequent Cycle: A labelled cycle h that has a similarity distance d_z, measured against a pre-specified hypothetical cycle referred to as the "zero-motif" (see Definition 9), where d_z occurs frequently in the collection D_z of similarity distances of cycles associated with the same class-label of h. A collection of candidate frequent cycles is then given by H', where $H' \subset H$.

Definition 8, Motif: A motif m is a sub-sequence held in some collection of sub-sequences that is representative of a class. The two strategies of determining whether a motif is representative or not which were explored with respect to this thesis are similarity- and frequency-based; more details relevant to each strategy is presented later on in this thesis.

Definition 9, Zero-motif: A hypothetical time series sub-sequence zm holding only zero values. The significance is that zero-motifs are used, as will become clear later in this thesis (Chapters 7 and 8), to identify motifs that cannot be frequent.

Definition 10, Top k Motifs: Given a set of motifs M, the k most representative motifs of a particular class. It is arguable as to which strategy (similarity-based, frequency-based or another strategy) is best for producing the most representative of the underlying class. The criteria for selecting the k motifs is still a research topic. The strategy for finding motifs and the criteria for selecting the top k motifs were studied with respect to the work presented later in this thesis; more specific details relevant to each is provided later in the thesis along with the corresponding algorithms.

Definition 11, Motif Bank: A set of motifs that can be used for Nearest Neighbour Classification.

4.3 Summary

This chapter has provided an overview of the formalism for the work presented in this thesis; a sequence of definitions was provided together with the notation used. Concepts like time series and motif are frequently used in the literature, but with a range of different interpretations. Some novel concepts, such as the "zero-motif" concept, were also defined in this chapter to facilitate the understanding of the following four chapters. As noted in Chapter 1, the various contributions of this thesis are presented in Chapters 5 to 8. In the following chapter, Chapter 5, a benchmark motif generation approach is presented with which the further approaches described in this thesis will be compared.

Table 4.1: Symbol table

p	A numeric value representing a point in a point (time) series.
P	A point (time) series comprising a sequence of points $\{p_1, p_2, \dots\}$ and, in the case of PCGs, consisting of a number of cardiac cycles.
P'	A pruned point series $\{p_1, p_2, \dots\}$, $P' \subset P$.
S	A segment representing some underlying sub-sequence of P, a tuple of the form $\langle S_p, S_C \rangle$, where S_p is the parent segment and S_C is a set of constituent sub-segments $\{S_{c_1}, S_{c_2}, \dots\}$.
$\overline{P}$	A segmented point series P, consisting of a sequence of segments $\{S_1, S_2, \dots\}$.
C	A set of classes $\{c_1, c_2, \dots\}$.
ω	A "window" size used to define the length of a point series sub-sequence.
q	A point series sub-sequence of length ω, $q \subset \mathbf{P}$, where $\mathbf{P}$ is P, P' or $\overline{P}$.
m	A motif, a sub-sequence q that is deemed (in some sense) to be representative of a class.
zm	The zero-motif, a hypothetical motif comprised of only zero values.
h	A heartbeat, a sub-sequence of P considered to form a cardiac cycle, $h \subset P$.
T	A set of point series and class-label pairs $\{\langle P_1, c_1 \rangle, \langle P_2, c_2 \rangle, \dots\}$.
Q	A set of sub-sequences of length ω, there will be a maximum of $x - \omega + 1$ sub-sequences in a points series of length x.
H	A set of cycle and class-label pairs $\{\langle h_1, c_1 \rangle, \langle h_2, c_2 \rangle, \dots\}$.
D_z	A set holding similarity values $\{d_{z_1}, d_{z_2}, \dots\}$, where d_{z_i} corresponds to the i^{th} element in a set of potential motifs.
H'	The set of pairs in H which appear within a specific range in a Gaussian Distribution, $H' \subset H$.
$\overline{H}$	A set of cycle and class-label pairs of a certain class-label c_i.
H''	The set of pairs in H' that are good class-label discriminators, $H'' \subset H'$.
f	The frequency with which a sub-sequence q occurs in a set of sub-sequences.
M	A set of motif and frequency value pairs $\{\langle m_1, f_1 \rangle, \langle m_2, f_2 \rangle, \dots\}$.
L	A set of motif and class-label pairs $\{\langle m_1, c_1 \rangle, \langle m_2, c_2 \rangle, \dots\}$, drawn from P, P' or $\overline{P}$.
D	A set of motif and class-label pairs $\{\langle m_1, c_1 \rangle, \langle m_2, c_2 \rangle, \dots\}$, drawn from T.
D'	The set of motifs in D that are good global class-label discriminators.
D''	The set of motifs in D that are not good global class-label discriminators, $D' \cap D'' = \emptyset$.
E	The energy of P comprising a sequence of values $\{e_1, e_2, \dots\}$.
E'	The standardised E.
V	The envelope of E' consisting of a number of oscillations.
X	A set of points in V that might leads to the boundaries of cardiac cycles h_i in P.
X'	A set of points in V that are anticipated to form the boundaries of h_i in P.
μ_d	The mean of D_z.
μ_e	The mean of E.
σ_d	The standard deviation of D_z.
σ_e	The standard deviation of E.
σ_m	A pre-specified frequency threshold.
α	A pre-specified oscillation-width threshold.
λ	A pre-specified similarity threshold for comparing two sub-sequences or motifs.
ϵ	A pre-specified threshold for the number of similar motifs selected from Q.
max	A pre-specified maximum size for a set of potential motifs.
k	A pre-specified threshold limiting the number of motifs in M to the k most frequently occurring motifs, $k < max$.
r	A pre-specified threshold for the number of references (iterations) in similarity calculation.
t	A dynamically computed threshold for detecting specific sequences in a time series.
t1,t2, t3,t4, t5	A pre-specified thresholds to define some shapes and types used in the context of segmentation.

Chapter 5

The MK Benchmark Phonocardiogram Classification Approach

5.1 Introduction

In this chapter, the first of the four PCG classification approaches considered is presented, the MK Benchmark PCG Classification approach. This is a benchmark approach, founded on the concept of motifs, that will be used as a vehicle with which to compare the alternative approaches presented later in this thesis. Any alternative approach, to be of merit, must be better than the benchmark presented in this chapter. The MK approach is founded on the MK algorithm first proposed by Abdullah **M**ueen and Eamonn **K**eogh [112] for the detection of motifs in time series; hence "MK". The idea underpinning the work presented in this chapter was to analyse the operation of the MK algorithm in the context of the PCG application domain. The intention was then to develop a number of refined variations according to the outcome of the analysis. The MK algorithm was selected because: (i) it is the first tractable exact motif discovery algorithm and (ii) when compared with a brute-force approach it is approximately three times faster. In the previous chapter, a number of definitions and notation relevant to work presented in this thesis were itemised; the relevant symbols with respect to this chapter are presented again in Table 5.1.

The main idea of the MK algorithm is to generate a number of random candidate motifs and select ϵ candidates that are most closely matched to each other. With respect to the implementation of the MK Benchmark approach considered in this chapter, Euclidean Distance was used to measure the similarity between any two sub-sequences q_i and q_j, expressed as $d(q_i, q_j)$. An "early abandonment" mechanism was also adopted in order to provide an early stop to computation where it was clear that the two sub-sequences under consideration were not similar and thus did not represent a motif.

Table 5.1: MK algorithm symbol table

p	A numeric value representing a point in a point (time) series.
P	A point (time) series comprising a sequence of points $\{p_1, p_2, \dots\}$ and, in the case of PCGs, consisting of a number of cardiac cycles.
C	A set of classes $\{c_1, c_2, \dots\}$.
ω	A "window" size used to define the length of a point series sub-sequence.
q	A point series sub-sequence of length ω, $q \subset P$.
m	A motif, a sub-sequence q that is deemed (in some sense) to be representative of a class.
T	A set of point series and class-label pairs $\{\langle P_1, c_1\rangle, \langle P_2, c_2\rangle, \dots\}$.
Q	A set of sub-sequences of length ω, there will be a maximum of $x - \omega + 1$ sub-sequences in a points series of length x.
L	A set of motif and class-label pairs $\{\langle m_1, c_1\rangle, \langle m_2, c_2\rangle, \dots\}$, drawn from P.
ϵ	A pre-specified threshold for the number of similar motifs selected from Q.
r	A pre-specified threshold for the number of references (iterations) in similarity calculation.

The remainder of this chapter is organised as follows. Section 5.2 gives a detailed description of the implemented MK algorithm. This is followed by an evaluation of the results obtained from experiments conducted using the proposed approach (Section 5.3). This is followed by some further discussion of the results in Section 5.4. This chapter is completed with a number of concluding remarks in Section 5.5.

5.2 MK Motif Discovery Algorithm

As discussed earlier in Chapter 2, a motif is a time series sub-sequence that is representative of a class, hence motifs have application with respect to time series classification. The advantage offered is that the entire time series does not need to be considered; an advantage of particular relevance with respect to very long time series such as PCG time series. Determining whether a motif is representative of a time series or not is still a research topic. Two strategies for determining whether a motif is representative or not are explored with respect to this thesis: (i) similarity- and (ii) frequency-based. With respect to the MK algorithm presented in this chapter, the similarity-based strategy was used. Motifs extracted based on similarity are determined using some similarity criteria, whereas motifs extracted based on frequency are determined using a frequency count criteria.

For the MK algorithm, as noted above, representativeness was measured from the similarity-based perspective. Given a set of sub-sequences Q, a motif is a sub-sequence $q \in Q$ that is most closely matched to $\epsilon - 1$ other sub-sequences in Q. The set of similar motifs, extracted from a single labelled time series, is then given by L; $L = \{\langle m_1, c_i\rangle, \langle m_2, c_i\rangle, \dots, \langle m_\epsilon, c_i\rangle\}$ where $c_i \in C$ is a class-label. In order to build a complete set of motifs ready for classification, and given a collection of time series T, each related to a class-label, the complete set of identified motifs is then given by $D = \bigcup_{i=1}^{z} L_i$, where z is the number of records (examples) in T.

A schematic of the MK approach for PCG classification is presented in Figure 5.1. In the figure, the training data set contains 9 labelled time series: of which 4, 3 and 2 belong to the classes blue, green and red respectively. The number of motifs at the end of the process is ϵ times the number of input time series; in other words, each time series in the "training data set" produces ϵ motifs. For the example given in the figure and throughout the thesis, $\epsilon = 2$ was used; so in the example, $2 \times 9 = 18$ motifs will be extracted and placed in the "motif bank". In the figure, the multiple arrows that enter and exit each step of the algorithm are simply to indicate that the time series could be processed in parallel.

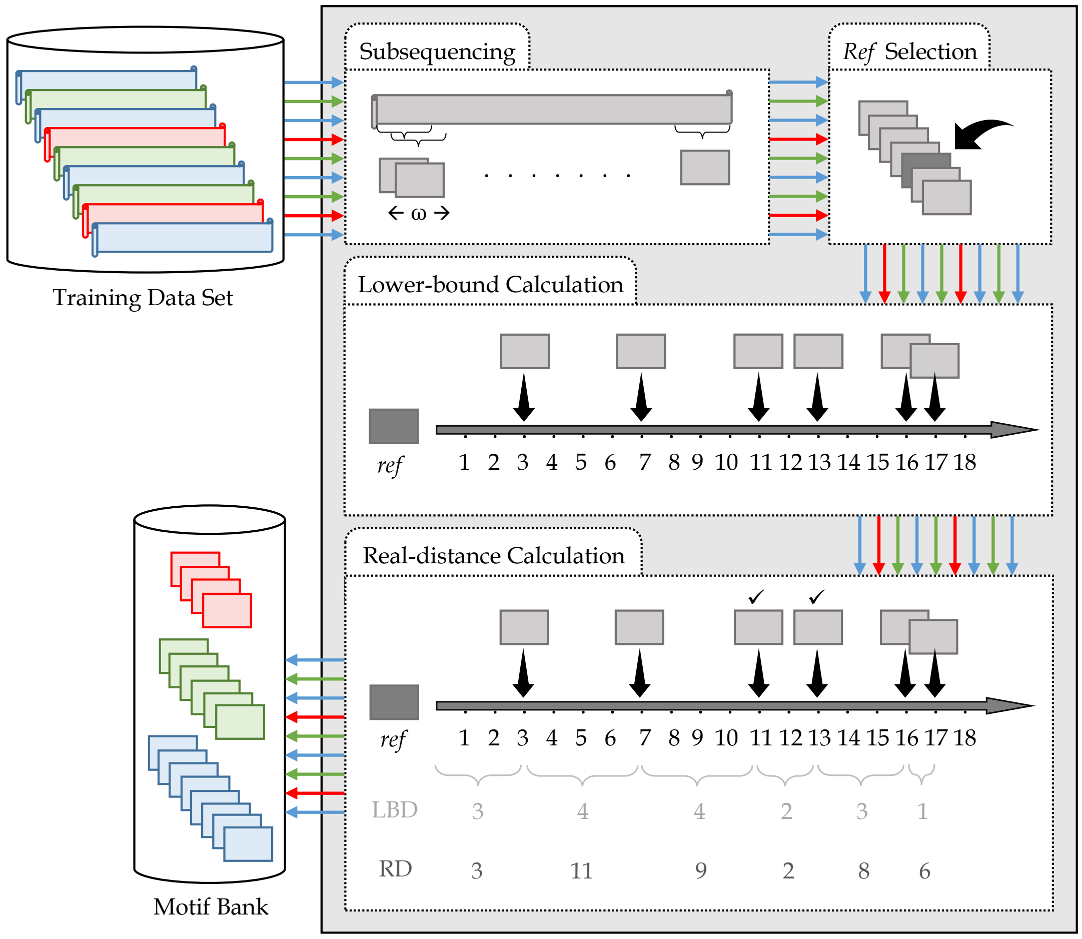


Figure 5.1: Schematic illustrating the high-level operation of the MK Benchmark PCG Classification approach

The first step shown in Figure 5.1 is "Subsequencing", the process of generating time series sub-sequences (candidate motifs). Each time series (PCG signal) in the input is divided into sub-sequences of some predefined length ω (the window size). In the next step, "*Ref* Selection", one of the sub-sequences is selected randomly as the "reference" time series sub-sequence (ref). In the next step, "Lower-bound Calculation" is conducted. The lower-bound in this context is the distance between the reference time series sub-sequence ref and all the rest time series sub-sequences in the input time series. The “lower-bound” is used to reduce the number of computations and consequently the required runtime. Thus, each sub-sequence, other than ref, is compared with ref and the distance (similarity) between them is calculated. The calculated distances can be conceptualised in terms of a linearisation as shown in the figure. These two steps, selecting a ref time series and calculating the lower-bound distances, are repeated r times. However, for simplicity, in the figure, $r = 1$ was used. The next step, “Real-distance Calculation”, is then to calculate the actual (“real”) distances between the sub-sequences so that the most similar pair can be identified. This is done by scanning the linearisation of the distances from left to right and selecting the pair of sub-sequences separated by the shortest real distance. These are considered to be the representative motifs (marked with a “✓” in the figure). The selected motifs are stored in a “motif bank” where then can be used in conjunction with some kind of Nearest Neighbour Classification (NNC) model to classify previously unseen time series. Further detail concerning the MK Benchmark PCG Classification approach proposed in this chapter is presented, in terms of pseudo code, in the following sub-section, Sub-section 5.2.1.

5.2.1 MK Benchmark PCG Classification Approach Pseudo Code

The previous section, Section 5.2, provided a high-level view of the MK Benchmark PCG Classification approach, the pseudo code for this algorithm is considered in this sub-section. The pseudo code for the parent process, the *Motif Discovery* process, is given in Algorithm 1. The input is: (i) a set T of point series and class pairs, (ii) a window size ω and (iii) a threshold r for the number of iterations. The output is a set L of identified motifs. The algorithm operates by processing each point series $P_i \in T$ in turn. First, a set Q is generated (line 2) comprised of the complete set of sub-sequences in P_i of length ω. This is then processed (line 3) to extract ϵ motifs, $\epsilon = 2$ was used as proposed in Mueen and Keogh. The two motifs, coupled with the class-label (line 4), are stored as two pairs in L. Once all the time series in T have been processed, the set L, holding the complete set of extracted motifs, is then returned (line 6) to be used as the “motif bank” for later classification.

Algorithm 1 MK Benchmark PCG Classification Approach

```
Require: $T$, $\omega$, $r$
Ensure: $L$
1: for $\forall\langle P_i, c_i\rangle \in T$ do
2:     $Q \leftarrow$ A set of sub-sequences of length $\omega$ in $P_i$
3:     $m_1, m_2 \leftarrow MK(Q, r)$
4:     $L \leftarrow L \cup \langle m_1, c_i\rangle \cup \langle m_2, c_i\rangle$
5: end for
6: return( $L$ )
```

The pseudo code for the MK algorithm, called from line 3 in the parent process, is given in Algorithm 2. The input is a set of sub-sequences Q and a number of iterations threshold r. The output is the two most similar motifs m_1 and m_2. The algorithm commences by defining three variables: (i) a two-dimensional array $Dist$, where the number of rows is equivalent to r and the number of columns is $|Q|$, (ii) an empty set SD to hold standard deviation values for each selected ref time series sub-sequence, and (iii) a variable *best-so-far* to hold the smallest distance between two time series sub-sequences as calculated so far, initialised with ∞ (infinity). Note that Q is anticipated to be large hence $Dist$ is anticipated to be large, and that r iterations will take place (lines 4 to 17). On each iteration, a reference sub-sequence ref_i is randomly selected from Q. The similarity between ref_i and all other sub-sequences in Q is then calculated and stored (line 8) in $Dist_i$, the i^{th} row in $Dist$. The smallest similarity value is maintained (lines 9 to 13). At the end of each iteration, the standard deviation of the row is computed and stored (line 16).

After r iterations, the two-dimensional array $Dist$ will be fully populated. The next step is to reorder the rows in $Dist$ according to the standard deviation values in SD in descending order (line 18). This reordering is done to facilitate searching of $Dist$. Standard deviation was used because the larger the standard deviation, the larger the lower-bound and hence the closer to the real distance. Thus, if the x^{th} element in SD is the highest value, the x^{th} row will be exchanged with the 1^{st} row; and so on. Next, the columns are reordered according to $Dist_1$ (the first row's) values in ascending order. This will reflect on the element order in Q (line 19). In other words, if the smallest number in the first row is in the x^{th} column, the whole x^{th} column will be exchanged with the 1^{st} column, hence in Q, q_x will be exchanged with q_1; and so on.

The next step is to find the smallest real distance (lines 20 to 44) using the lower-bound distances stored in $Dist$. The process initialises two variables for the search: $offset$ which is an incremental number initialised with 0, and *abandon* which is a Boolean flag, for maintaining the "early abandonment" concept, initialised with $false$. The loop stops searching when *abandon* = *true* (line 24). In this loop, there are two nested loops to process $Dist$; the outer loop processes the columns (line 25) whilst the inner loop (line 27) processes the rows within each column. Each cell will be compared with its neighbour in the following column (lines 28 and 29). If the lower-bound distance between them is less than the *best-so-far* distance, the real distance is calculated; if the real distance is less than the *best-so-far* distance, the latter will be updated with the real distance (line 38). The two potential motifs are stored in m_1 and m_2 (lines 39 and 40). Once a cell holds a number greater than the *best-so-far* distance, the search will be terminated. The reason for the termination is that having a number greater than *best-so-far* means that the rest of the array is certainly greater; this is why the array was sorted in ascending order and consequently provides an efficiency gain. The two motifs, m_1 and m_2, with the best (smallest) associated similarity distance value are then returned (line 45).

Algorithm 2 MK

```
Require: $Q$, $r$
Ensure: $m_1$, $m_2$
1: $Dist \leftarrow \emptyset$, An $r \times |Q|$ array to hold the distances between sub-sequences
2: $SD \leftarrow \emptyset$, A set to hold standard deviation values
3: $best\_so\_far \leftarrow \infty$, A value of the lowest distance between two sub-sequences so far
4: for $i = 1$ to $r$ do
5:   $ref_i \leftarrow$ A randomly chosen sub-sequence from $Q$
6:   for $j = 1$ to $|Q|$ do
7:     if $ref_i$ and $q_j$ not a trivial match then
8:       $Dist_{i,j} \leftarrow d(ref_i, q_j)$
9:       if $Dist_{i,j} < best\_so\_far$ then
10:        $best\_so\_far \leftarrow Dist_{i,j}$
11:        $m_1 \leftarrow ref_i$
12:        $m_2 \leftarrow q_j$
13:      end if
14:    end if
15:  end for
16:  $SD \leftarrow SD \cup standard\_deviation(Dist_i)$
17: end for
18: Reorder $Dist$ rows in descending order according to standard deviation values in $SD$
19: Reorder $Dist$ columns in ascending order according to $Dist_1$ (the first row in $Dist$)
    values, given that this will reflect on the element order in $Q$
20: $offset \leftarrow 0$, An incremental number
21: $abandon \leftarrow$ false, A flag for terminating the computations
22: while $abandon =$ false do
23:   $offset \leftarrow offset + 1$
24:   $abandon \leftarrow$ true
25:   for $j = 1$ to $|Q|$ do
26:     $reject \leftarrow$ false
27:     for $i = 1$ to $r$ do
28:       $lower\_bound \leftarrow |Dist_{i,j} - Dist_{i,(j+offset)}|$
29:       if $lower\_bound > best\_so\_far$ then
30:         $reject \leftarrow$ true
31:         break
32:       else if $i = 1$ then
33:         $abandon \leftarrow$ false
34:       end if
35:     end for
36:     if $reject =$ false then
37:       if $d(q_j, q_{(j+offset)}) < best\_so\_far$ then
38:         $best\_so\_far \leftarrow d(q_j, q_{(j+offset)})$
39:         $m_1 \leftarrow q_j$
40:         $m_2 \leftarrow q_{(j+offset)}$
41:       end if
42:     end if
43:   end for
44: end while
45: return( $m_1$ , $m_2$ )
```

5.3 Evaluation

In the previous section, Section 5.2, the operation of the MK Benchmark Phonocardiogram Classification approach was presented. This section considers the evaluation results obtained with respect to a set of experiments undertaken, using the collected evaluation data set (see Chapter 3), to analyse the operation of the proposed approach. The evaluation metrics recorded were accuracy (acc), precision (prec), recall (rec), F-score (f-s) and runtime. Five-fold cross-validation was adopted and the data was statically stratified with respect to all the reported experiments. The evaluation was conducted using the Java programming language and run on an iMac Pro (2017) computer with 8-Cores, 3.2GHz Intel Xeon W CPU and 19MB RAM. The objectives of the evaluation were:

Objective 1, Operational Analysis: To investigate the operation of the proposed approach.

Objective 2, Most Appropriate Representation: To identify the most appropriate representation, Amplitude- or Sample-PCG, in terms of computational efficiency (runtime).

Objective 3, Most Appropriate Classifier: To identify the most appropriate classification model. Two were considered: (i) the well-known Nearest Neighbour Classification (NNC) [38, 87] and (ii) Smallest Average Classification (SAC), a variation of NNC developed by the author.

Objective 4, Most Appropriate Parameter Settings: To establish the most appropriate parameter settings for: (i) ω, the size of the candidate motifs and (ii) r, the number of the references.

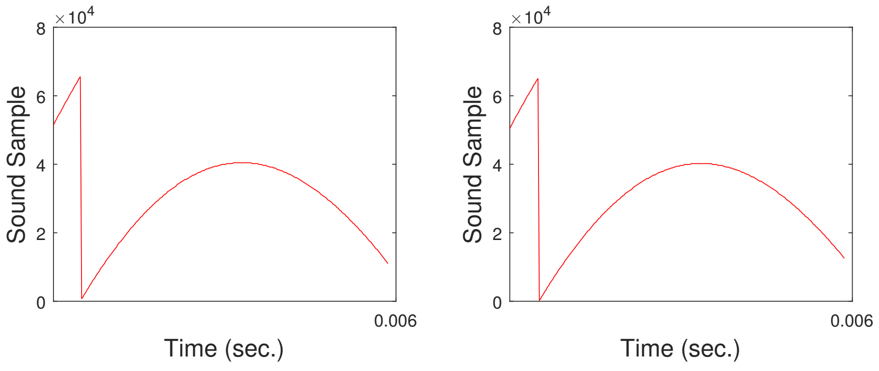
Each of the above objectives are considered in turn in the following four sub-sections, Sub-sections 5.3.1 to 5.3.4.

5.3.1 Operational Analysis

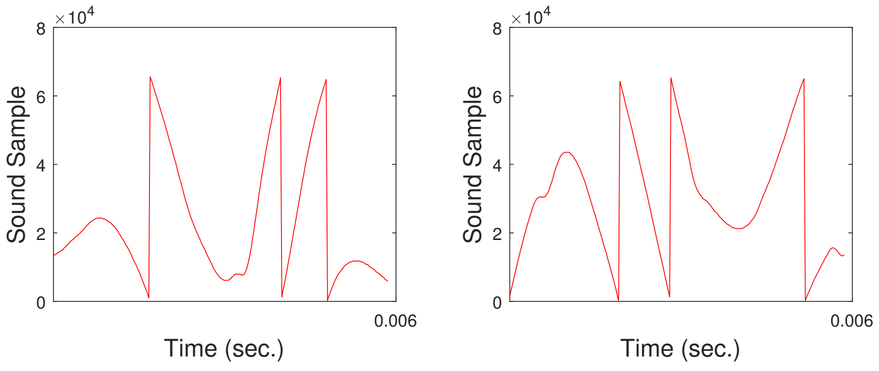
In this sub-section, the operation of the application of the proposed MK Benchmark Phonocardiogram Classification approach to PCG data is considered. There has been no previous work, to the best knowledge of the author, that uses motifs to address the PCG classification problem.

Figures 5.2 and 5.3 show example Sample- and Amplitude-PCG motifs (respectively) for each of the four classes considered with respect to the PCG evaluation data collection used throughout this thesis. These examples were all generated using the proposed approach with $\epsilon = 2$.

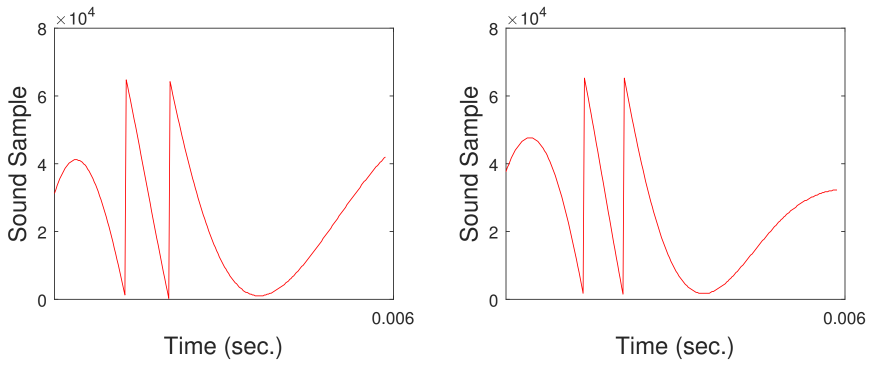
Using five cross-validation, the Sample-PCG representation and NNC, the best and worst accuracies recorded were 71.1% and 61.7%. The associated precision, recall and F-score values were 0.225, 0.351 and 0.270 for the best accuracy, and 0.097, 0.146 and 0.116



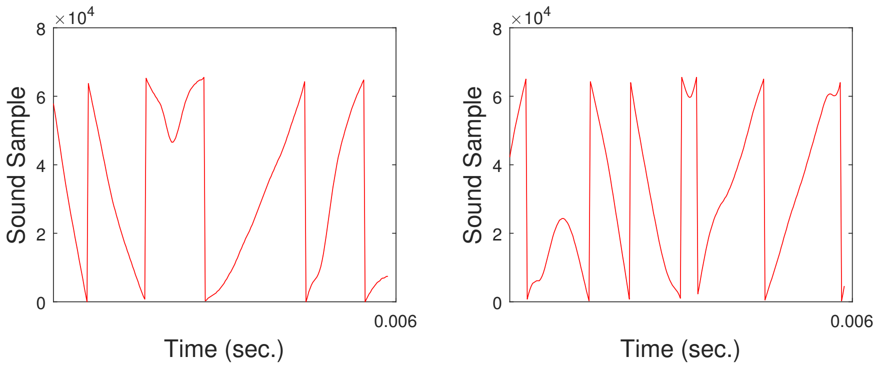
(a) Class# 1



(b) Class# 2

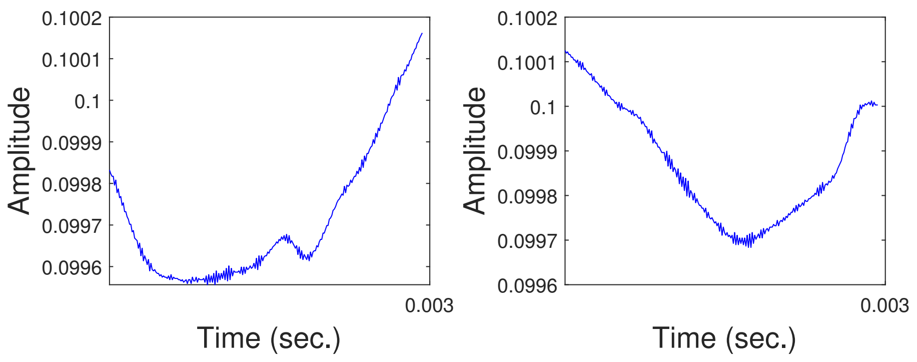


(c) Class# 3

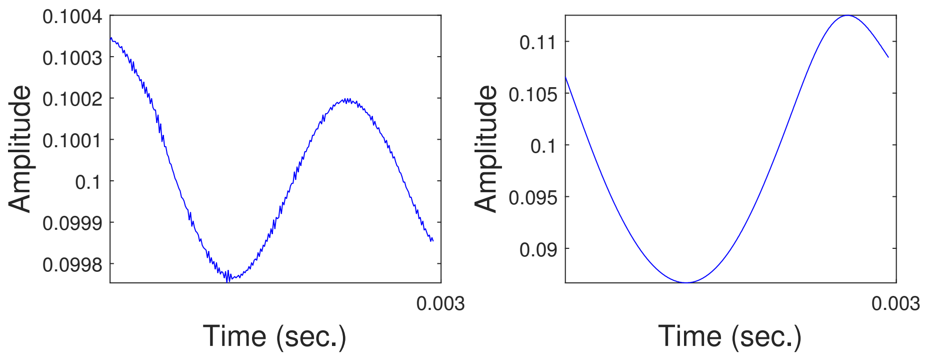


(d) Class# 4

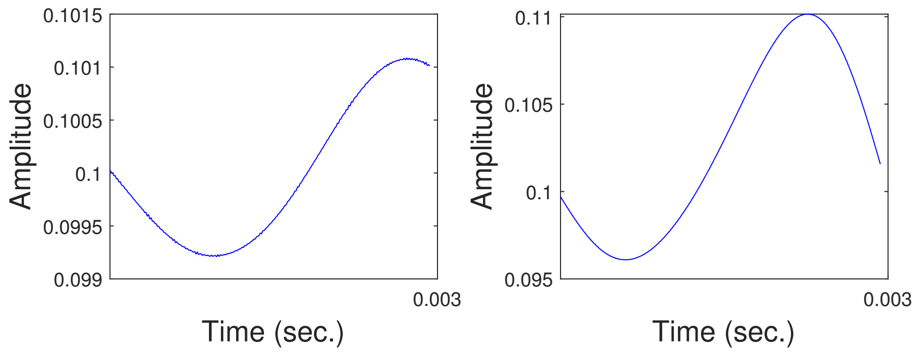
Figure 5.2: Examples of pairs of Sample-PCG motifs, for each class in the PCG evaluation data, generated using the proposed MK Benchmark PCG Classification approach



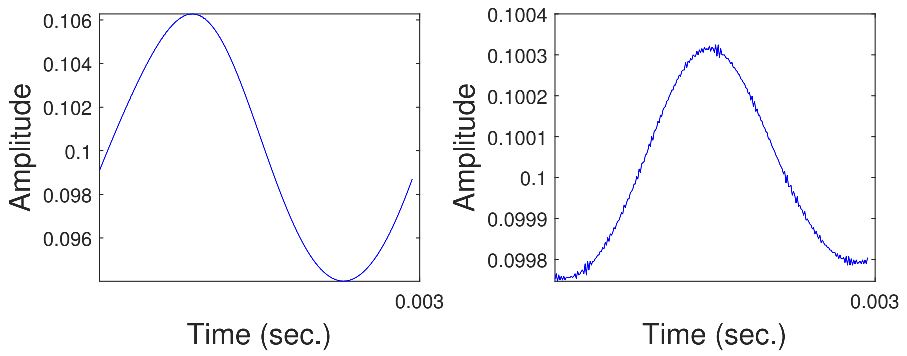
(a) Class# 1



(b) Class# 2



(c) Class# 3



(d) Class# 4

Figure 5.3: Examples of pairs of Amplitude-PCG motifs, for each class in the PCG evaluation data, generated using the proposed MK Benchmark PCG Classification approach

for the worst. The recorded standard deviation for the classification model was good (0.05 on an average). However, the average runtime to extract $\epsilon = 2$ motifs from one record was more than 2 hours. The total time to process the whole data set was more than 5 days, on average, for each experiment (a set of nine experiments was conducted to determine the best values for the MK variables, more details of which are given in Sub-section 5.3.4). Using the Amplitude-PCG representation, even more execution time was required.

5.3.2 Most Appropriate Representation

As already noted, two PCG data representations were considered: Amplitude (see Figure 2.3(a)) and Sample (see Figure 2.3(b)). In this sub-section, these two representations are compared. From the experiments, as already noted in the previous sub-section, it was found that the runtime associated with the Amplitude-PCG representation was much greater than that associated with the Sample-PCG representation. The recorded runtime results are presented in Table 5.2 for a range of values of ω and r. From the table, it can be clearly seen that the MK algorithm coupled with the Amplitude-PCG representation required approximately 18 hours to extract $\epsilon = 2$ motifs from each record. It took more than 44 days to extract the motifs from the entire data set just for one five-fold cross-validation experiment. Since the plan was to conduct nine experiments, it was anticipated that MK would probably require more time as the parameter values increased. For confirmation, the second experiment (out of the planned nine) was conducted; it was found that it needed more than 86 days. Therefore, it was decided to not continue the sequence of experiments using the MK algorithm with the Amplitude-PCG representation as indicated in Table 5.2 using "N/A".

Table 5.2: Recorded runtimes (hh:mm:ss format) for MK Benchmark PCG Classification approach with respect to Amplitude- and Sample-PCG representation

ω	r	Sample-PCGs	Amplitude-PCGs
250	2	00:42:39	17:57:54
	4	00:51:18	35:16:31
	6	00:59:25	N/A
500	2	00:51:11	N/A
	4	01:02:14	N/A
	6	01:26:27	N/A
750	2	04:10:22	N/A
	4	04:32:22	N/A
	6	05:17:44	N/A
Average		02:12:38	26:37:12

The Sample-PCG representation needed significantly less runtime (relatively). Approximately, ninth time of that required when using the Amplitude-PCG representation. Thus, the Sample-PCG representation was adopted with respect to the remainder of the evaluations reported on in this chapter.

5.3.3 Most Appropriate Classifier

Various number of classification models can be applied with respect to the proposed MK Benchmark PCG Classification approach, although Nearest Neighbour Classification (NNC) seemed the most obvious choice, backed by the observation that this style of classification model is frequently used in the context of time series classification [38, 87, 112]. As noted above, two classification models were considered with respect to the evaluation presented here: NNC and SAC. In this sub-section, each of these models is first briefly described, before considering their performance in terms of accuracy, precision, recall and F-score. For the evaluation reported here using NNC, $k_{nnc} = 1$ was used although, as discussed in the following sub-section, no performance difference was found when considering a higher value for k_{nnc} such as $k_{nnc} = 3$.

The well-known NNC model operates by finding the most similar existing (labelled) motif to a previously unseen motif to be classified. NNC uses a voting mechanism; the number of votes is specified by a variable k_{nnc}. When k_{nnc} is greater than one, some mechanism is required to resolve conflicts. Similarity is measured using some distance functions [33]. For continuous variables (as in this thesis), functions such as Euclidean, Manhattan and Minkowski distance may be used; Euclidean Distance was adopted with respect to the evaluation presented here because it is frequently used in time series analysis [73, 160, 182, 185, 198] and specifically with respect to motif discovery algorithms [21, 36, 77, 112].

The SAC classifier, unlike the NNC classifier, takes into consideration the similarity between the new motif to be classified and all motifs for each class in the "motif bank". The algorithm operates as follows. Distances are calculated between the new motif and all motifs for each class. Then, for every class, the average of the distances is computed. The new motif will then be classified with the class-label that features the smallest average. For computing the distance, Euclidean Distance was again used.

The performance results, obtained using NNC and SAC, are presented in Table 5.3. As can be seen from the table, the accuracy using NNC was better than that obtained using SAC. There is little to distinguish between the precision and recall results obtained. Therefore, with respect to the rest of the evaluation presented in this section, only results obtained using NNC are considered.

5.3.4 Most Appropriate Parameter Settings

The proposed MK Benchmark PCG Classification approach uses two parameters: ω to define the size (length) of the candidate motifs, and r to determine the number of references in similarity calculation. Experiments were conducted using a number of alternative values for these parameters; $\omega = \{250, 500, 750\}$ and $r = \{2, 4, 6\}$. However, it should be noted that it was reported in [112] that the value of r was not critical and that any value greater than 5 makes little difference. The three values for r and three values for ω combined to give nine separate combinations; the nine experiments referred to in Sub-section 5.3.1.

Table 5.3: Comparison of classification performance using NNC and SAC operating with motifs generated using the proposed MK Benchmark PCG Classification approach and the Sample-PCG representation

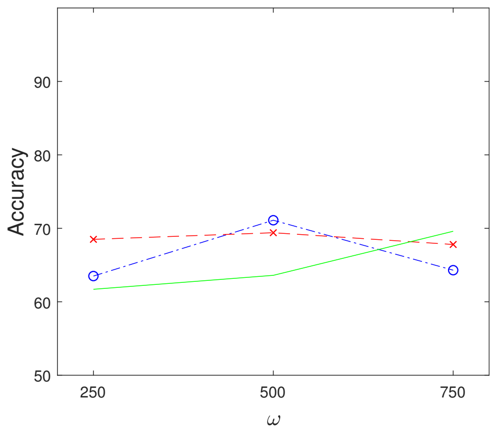
ω	r	NNC ($k_{nnc} = 1$ or 3)				SAC			
		Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
250	2	**0.635**	0.133	**0.206**	0.159	0.602	**0.166**	0.181	**0.163**
	4	**0.685**	**0.148**	0.226	0.175	0.592	**0.148**	**0.316**	**0.191**
	6	**0.617**	0.097	0.146	0.116	0.601	**0.142**	**0.234**	**0.171**
500	2	**0.711**	0.225	0.351	0.270	0.627	**0.276**	**0.378**	**0.307**
	4	**0.694**	**0.174**	**0.291**	**0.213**	0.601	0.147	0.269	0.187
	6	**0.636**	**0.195**	0.179	**0.177**	0.567	0.136	**0.201**	0.145
750	2	**0.643**	0.154	0.183	0.160	0.600	**0.195**	**0.264**	**0.190**
	4	**0.678**	**0.189**	**0.286**	**0.224**	0.559	0.146	0.153	0.126
	6	**0.696**	**0.290**	**0.296**	**0.282**	0.533	0.086	0.068	0.063

The criteria used for the evaluation was classification performance using NNC, because earlier experiments, reported above, had demonstrated this to be a good choice.

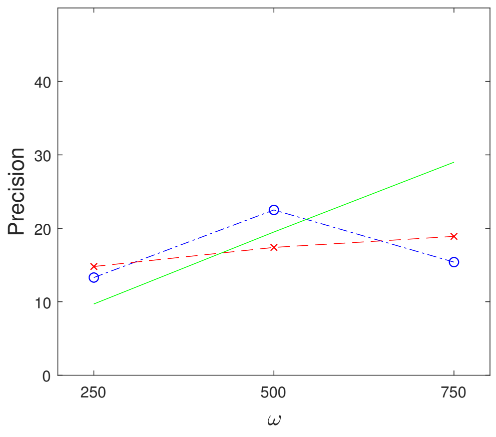
Since NNC classification uses a parameter k_{nnc}, number of votes, this also needed to be defined. For the experiments reported here, $k_{nnc} = 1$ and $k_{nnc} = 3$ were used. Comparison of the results demonstrated that both values produced exactly the same outcome. Inspection of the results indicated that the agreement was consequent to either: (i) the second and/or third nearest neighbours (when $k_{nnc} = 3$) agreeing with the first so as to give the same results as when $k_{nnc} = 1$ or (ii) the three nearest neighbours giving three different class-labels therefore the "closest" class-label was selected, the class-label when $k_{nnc} = 1$. No cases were found where the second and third nearest neighbours agreed on a class-label that was not compatible with the first nearest neighbour's class-label.

Figure 5.4 presents the results obtained in terms of a sequence of five plots. The values for the first four plots (accuracy, precision, recall and F-score) were given earlier in Table 5.3. The best accuracy (with respect to the conducted experiments) was obtained using $\omega = 500$ and $r = 2$ (71.1%). The lowest accuracy score was obtained using $\omega = 750$ and $r = 6$. Inspection of the figure indicates that the accuracy, precision, recall and F-score plots display similar behaviour; this is to be anticipated as these four metrics are related to one another.

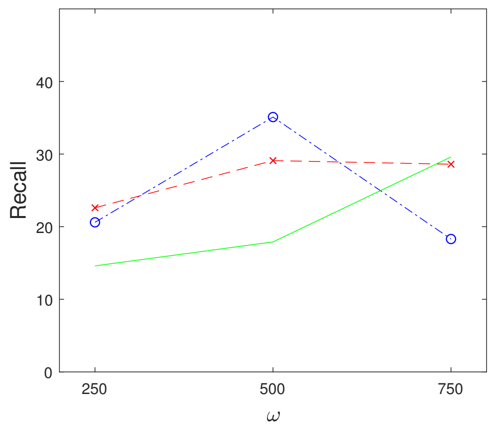
The recorded runtimes were presented previously in Table 5.2, but are shown in graph form in Figure 5.4(e) in hh:mm format. The runtime required by the proposed approach using the Sample-PCG representation and $k_{nnc} = 1$ for the NNC model, to complete all the nine experiments, was roughly 48 days and 21 hours. The shortest runtime obtained (about 42 minutes) was when the minimum values for the parameters were used, whilst the longest time (about 5 hours) was when the maximum values for the parameters were considered. From Figure 5.4(e), it can be seen that the runtime increases proportionally as the values associated with the ω and r parameter values increases.



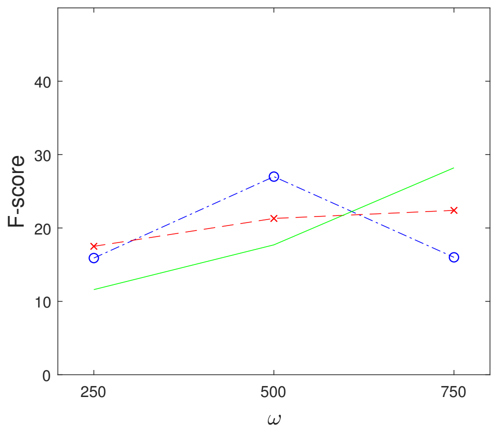
(a) Accuracy



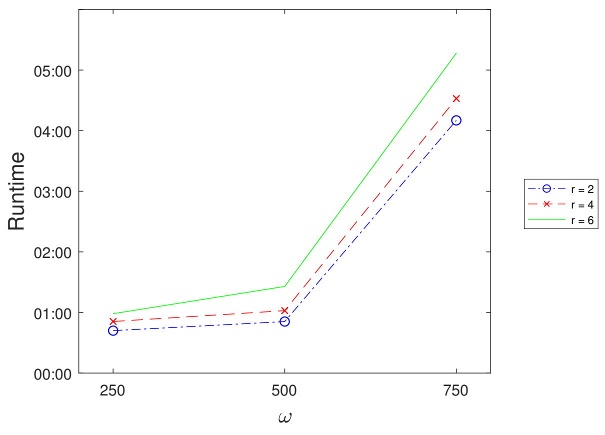
(b) Precision



(c) Recall



(d) F-score



(e) Runtime

Figure 5.4: Sequence of plots recording the results obtained from the evaluation of the proposed MK Benchmark PCG Classification using the Sample-PCG representation and NNC with $k_{nnc} = 1$

5.4 Discussion

From Figure 5.4(e), the required runtime for the MK Benchmark approach to discover the best motif pair was significantly high; arguably unacceptably high. There does seem to be an identifiable "pattern" as can be seen from inspection of the figure; the greater the value of r, the greater the runtime. However, the greatest influence on runtime was the selection of the value for the parameter ω. The runtime increased dramatically given $\omega = 750$, regardless of the opposite being reported in [112]. In [112], it was claimed that the more references there are, the lower the runtime, till $r = 60$; although from $r = 5$ onwards, the effect was negligible. The evaluation of the MK algorithm as reported in [112] was conducted using a set of 30,000 sub-sequences of length 1,024 and, it was claimed, the effect of r was consistent regardless of the value of ω, number of sub-sequences and data type. The PCG signals used in this thesis yield approximately 25 million sub-sequences; this could be the reason for the different results from what was reported in [112].

With respect to accuracy, Figure 5.4(a) demonstrated that there was no clear relationship between ω and accuracy. Moreover, given that it has been established that the effect of r is negligible, it would be reasonable to anticipate consistent results using the same ω value. However, each of the three separate experiments that used the same ω value gave non-convergent results. This may be due to the randomness in choosing the references in the MK algorithm.

Based on the above, it was conjectured that the limited performance efficiency was a consequence of the high data volumes (extracting data points from PCG files yields more than 25.5 million data points). Therefore, it was suggested that the PCG point series should be pre-processed prior to any further processing being conducted. This might be done by changing the representation of the point series so that the size of the point series is reduced, but in such a way that salient characteristics are preserved. It was also conjectured that this idea may contribute to classification effectiveness as it would serve to reduce runtime and remove unimportant features. These ideas are explored further later in this thesis.

5.5 Summary

This chapter has presented an approach to the automated analysis of PCG data founded on the idea of time series analysis; more specifically, using the established MK algorithm. The objective of this chapter was to identify a mechanism for determining indicative patterns (motifs) that can be used to classify PCG data, in an efficient and effective manner. The performance of the proposed approach was acceptable to some extent (best accuracy values of 71.1% were obtained) but the runtime was an issue. To address this runtime problem, the next chapter proposes an alternative PCG classification approach founded on the idea of data segmentation so as to reduce the overall size of the data; given that the data size is usually the main contributor to the adverse runtimes, as in the case of the recorded runtime results presented in this chapter.

Chapter 6

The PCG_{seg} Classification Approach for Phonocardiogram Classification

6.1 Introduction

In the previous chapter the MK Benchmark PCG Classification approach was presented. A particular disadvantage in the way that the Benchmark approach was applied was the excessive runtime, resulting from the size of the PCG time series considered. It was therefore suggested in the previous chapter that, in order to reduce the size of the PCG data, an alternative representation was required to the Amplitude- and Sample-based representations considered. The idea presented in this chapter, to address the runtime issue as identified in the previous chapter, was to pre-processing the PCG data so as to reduce the overall size of the time series. This chapter presents the PCG_{seg} Classification approach; central to the approach is a novel segmentation algorithm, the PCG_{seg} algorithm, from which the approach gets its name, to be applied to PCG data prior to motif discovery. For the motif discovery, it is proposed that the MK motif discovery algorithm is again used, but with some modification to the similarity function to suit segmented point series. The main objective of the work presented in this chapter was thus to reduce the size of the PCG time series, using some form of segmentation, so that a useful classification model could be generated in an efficient manner. The Sample-based PCG representation was adopted as the input representation, because work in the previous chapter had indicated that this was a more appropriate representation than the alternative Amplitude-based representation also considered in the previous chapter. The relevant notation and symbols used with respect to this chapter are given in Table 6.1, taken from Chapter 4.

The remainder of this chapter is organised as follows. An overview of PCG_{seg} Classification approach is presented in Section 6.2. The two main components of the approach are then discussed in further detail in Sections 6.3 and 6.4. Section 6.3 gives a detailed

Table 6.1: PCG_{seg} Classification approach symbol table

p	A numeric value representing a point in a point (time) series.
P	A point (time) series comprising a sequence of points $\{p_1, p_2, \dots\}$ and, in the case of PCGs, consisting of a number of cardiac cycles.
S	A segment representing some underlying sub-sequence of P, a tuple of the form $\langle S_p, S_C \rangle$, where S_p is the parent segment and S_C is a set of constituent sub-segments $\{S_{c_1}, S_{c_2}, \dots\}$.
$\overline{P}$	A segmented point series P, consisting of a sequence of segments $\{S_1, S_2, \dots\}$.
C	A set of classes $\{c_1, c_2, \dots\}$.
ω	A "window" size used to define the length of a point series subsequence.
m	A motif, a subsequence q that is deemed (in some sense) to be representative of a class.
T	A set of point series and class-label pairs $\{\langle P_1, c_1 \rangle, \langle P_2, c_2 \rangle, \dots\}$.
Q	A set of subsequences of length ω, there will be a maximum of $x - \omega + 1$ subsequences in a points series of length x.
L	A set of motif and class-label pairs $\{\langle m_1, c_1 \rangle, \langle m_2, c_2 \rangle, \dots\}$, drawn from $\overline{P}$.
ϵ	A pre-specified threshold for the number of similar motifs selected from Q.
r	A pre-specified threshold for the number of references (iterations) in similarity calculation.
$t1$	A pre-specified threshold to define a vertical shape (used in the context of segmentation).
$t2$	A pre-specified threshold to define a slant shape (used in the context of segmentation).
$t3$	A pre-specified threshold to define a dome shape (used in the context of segmentation).
$t4$	A pre-specified threshold to define the type of dome and flat shapes (used in the context of segmentation).
$t5$	A pre-specified threshold to define the sub-segment type (used in the context of segmentation).

description of the proposed PCG_{seg} segmentation technique. Section 6.4 presents the motif discovery approach used with the segmented data. In Section 6.5 an evaluation of the results obtained from experiments conducted using the proposed approach is given. This is followed by some further discussion of the results in Section 6.6. The chapter is concluded with a summary in Section 6.7.

6.2 The PCG_{seg} Classification Approach

This section presents the proposed PCG_{seg} Classification approach. A schematic of the proposed approach, the PCG_{seg} segmentation technique coupled with the MK algorithm [112] for PCG classification, is presented in Figure 6.1. As in the case of Figure 5.1, in Figure 6.1 the example training data set contains 9 labelled time series: 4, 3 and 2 time series belong to the classes blue, green and red respectively. Because the MK algorithm suggests selecting $\epsilon = 2$ motifs [112], which is the number used throughout this chapter, the number of motifs at the end of the process is $\epsilon = 2$ times the number of input time series. For the example given in the figure, $2 \times 9 = 18$ motifs will be extracted and placed in the "motif bank". In the figure, the multiple arrows that enter and exit each step of the algorithm are intended to indicate that the time series could be processed in parallel.

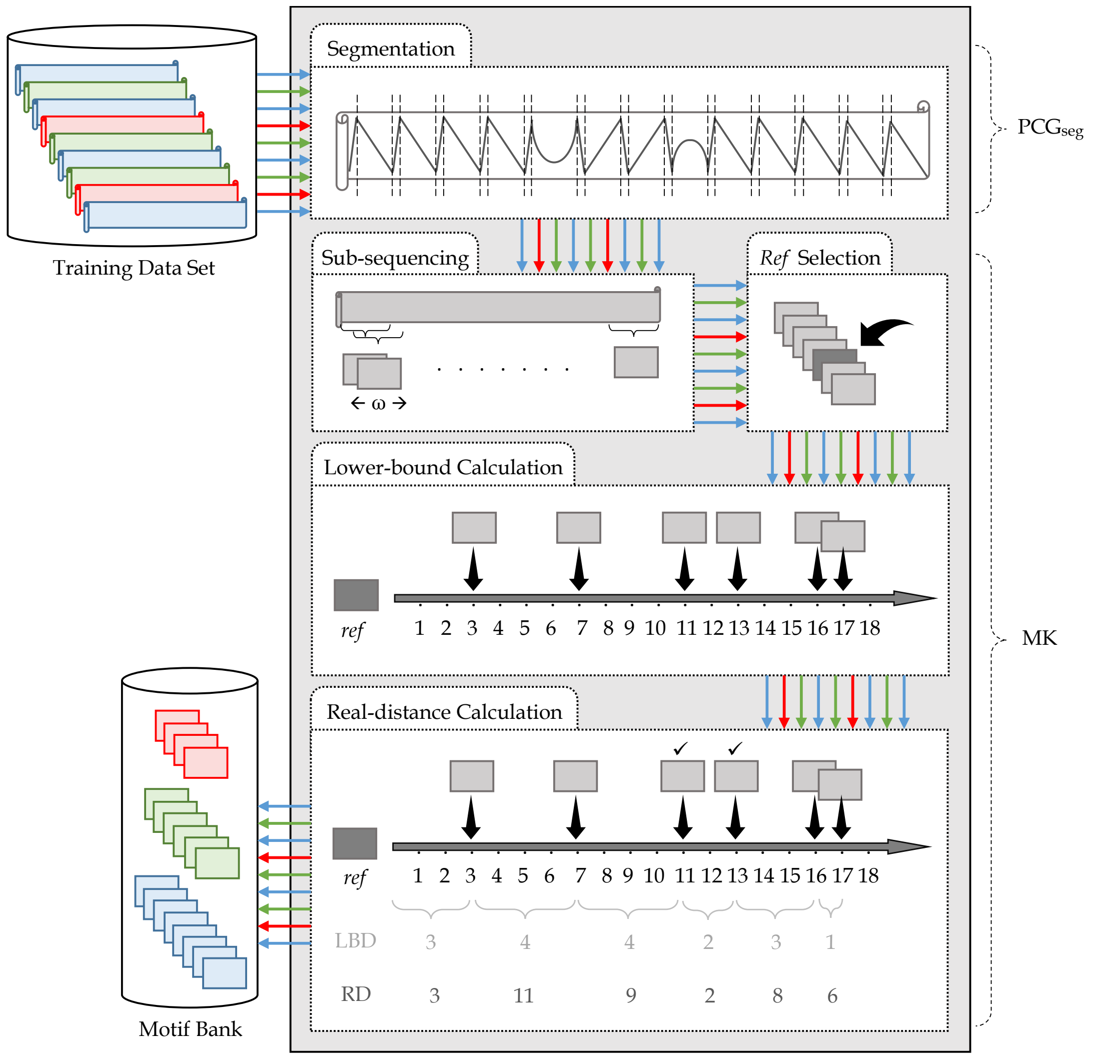


Figure 6.1: Schematic illustrating the basic operations of the PCG$_{seg}$ Classification approach

In more detail, the PCG$_{seg}$ Classification approach processes each time series by detecting its segments and sub-segments. The segmented time series are then passed to the "Sub-sequencing" (candidate motif generation) step. Each segmented time series is divided into sub-sequences of some predefined length ω, measured in terms of the number of segments. Next, a random sub-sequence is selected as the "reference" sub-sequence (ref). The sub-sequences are then passed to the "Lower-bound Calculation" step where the Lower-Bound Distances (LBDs), between the ref and the remaining sub-sequences in the input time series, are calculated. The calculated distances can be conceptualised in terms of a linearisation as shown in the figure. These two steps, "*Ref* Selection" and "Lower-bound Calculation", are repeated r times. In the figure, for simplicity, $r = 1$ was used. The next step, is then to calculate the actual Real Distances (RDs) between the sub-sequences so that the most similar pair can be identified. This is done by scanning the linearisation of the distances from left to right and selecting the pair of sub-sequences separated by the shortest real distance. These are considered to be the representative motifs (marked with a "✓" in the figure). The selected motifs are stored in a "motif bank" where they can be used in conjunction with some kind of Nearest Neighbour Classification (NNC) model to classify previously unseen time series.

The pseudo code for the parent PCG_{seg} Classification approach is given in Algorithm 3. The inputs are: (i) a set T of point series and class pairs, (ii) a set of five thresholds for the segmentation algorithm, $\{t1, t2, t3, t4, t5\}$ (as will be discussed later) and (iii) the two variables needed by the MK algorithm: a window size ω and a threshold r for the number of iterations. The output is a set L of identified motifs. The algorithm operates by processing each point series $P_i \in T$ in turn. First, $\overline{P_i}$ is generated by segmenting P_i (line 2). Then, a set Q is generated with the complete set of sub-sequences in $\overline{P_i}$ of length ω (line 3). This is then processed (line 4) to extract $\epsilon = 2$ motifs; m_1 and m_2. The two motifs coupled with the class-label (line 5) are stored as two pairs in L. Once all the time series in T have been processed, the set L will hold the complete set of extracted motifs, this is then returned (line 7) to be used as the "motif bank" for later classification.

Algorithm 3 PCG_{seg} Classification Approach

```
Require: $T$, $\{t1, t2, t3, t4, t5\}$, $\omega$, $r$
Ensure: $L$
1: for $\forall \langle P_i, c_i \rangle \in T$ do
2:     $\overline{P_i} \leftarrow PCG_{seg}(P, t1, t2, t3, t4, t5)$
3:     $Q \leftarrow$ A set of sub-sequences of length $\omega$ in $\overline{P_i}$
4:     $m_1, m_2 \leftarrow MK(Q, r)$
5:     $L \leftarrow L \cup \langle m_1, c_i \rangle \cup \langle m_2, c_i \rangle$
6: end for
7: return( $L$ )
```

From Algorithm 3, it can be seen that there are two function calls. The first call (line 2) is for the segmentation pre-processing, PCG_{seg}, this is presented in the following section, Section 6.3. The second function call (line 4) is for the motif discovery super-process, the adjusted MK algorithm discussed in further detail in Section 6.4.

6.3 Segmenting Algorithm

A number of existing segmentation methods, taken from previous work, were presented in Chapter 2. The observation was made that these methods were general-purpose methods, not well-suited to the Sample-PCG signals, the used representation with respect to this chapter. As a consequence, it was proposed that a bespoke segmentation technique, specifically directed at the segmentation of PCG time series, given in terms of a Sample-based representation, was required. This section therefore presents the proposed Sample-PCG segmentation technique, the PCG$_{seg}$ technique.

As in the case of existing segmentation mechanisms, the idea is still to reduce the number of elements in the point series in question, but also to produce a representation that supports motif-based PCG time series classification. The anticipation was that this would be more efficient than comparable motif-based methods, of the form considered in the previous chapter, that do not use segmentation. Furthermore, it was anticipated that PCG$_{seg}$ would improve the accuracy of PCG classification compared to the results obtained using the unsegmented Benchmark approach presented in the previous chapter.

The reminder of this section is divided into three sub-sections. Sub-section 6.3.1 provides some details covering the theoretical underpinning for the proposed PCG$_{seg}$ approach. Sub-section 6.3.2 then presents the PCG$_{seg}$ segmentation technique. Sub-section 6.3.3 provides an illustration of the operation of the proposed PCG$_{seg}$ algorithm.

6.3.1 Segments and Sub-segments

The proposed PCG$_{seg}$ algorithm is underpinned by the idea of capturing the “shapes” that exist in a PCG sequence. The idea is illustrated in Figure 6.2. From the figure, it can be seen that a PCG sequence, in the form of the Sample-based representation, can be conceptualised in terms of a series of shapes and sub-shapes (segments and sub-segments). In the figure, four distinct shapes can be identified: (i) slant (black), (ii) vertical (red), (iii) dome (green) and (iv) flat (cyan). It is also important to note that the vertical shape always occurs between any two other shapes; in other words, it is a separator and this feature of the Sample-PCG data of interest is used within the proposed PCG$_{seg}$ mechanism to identify the start and end points of segments.

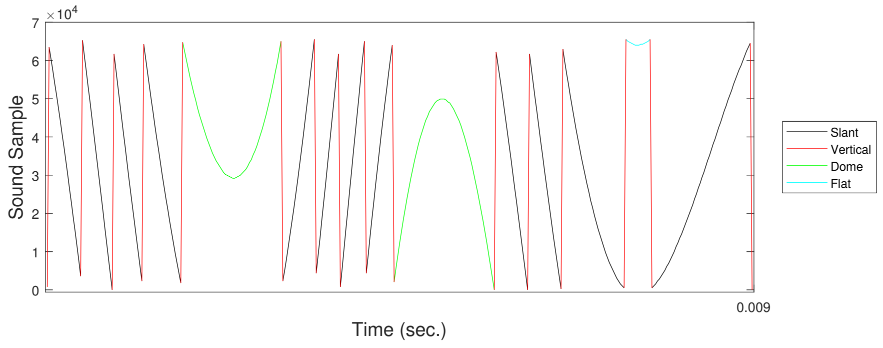
As already noted in the formalism chapter, Chapter 4, a segment is defined by a tuple of the form $\langle shape, type, length \rangle$. The possible values for the *shape* variable are: {vertical, slant, dome, flat} as illustrated in Figure 6.2. Each segment is defined in terms of a conceptual Minimum Bounding Box (MBB) surrounding it. The x-dimension of the MBB surrounding a segment corresponds to the value of the *length* variable associated with the segment. More specifically, each *shape* is defined as follows:

- **Slant**: A slant shape comprises a sequence of three or more points ($length \geq 2$) such that the start and end points are at opposite corners of the MBB and the difference between the start and end point values is greater than a threshold $t1$.

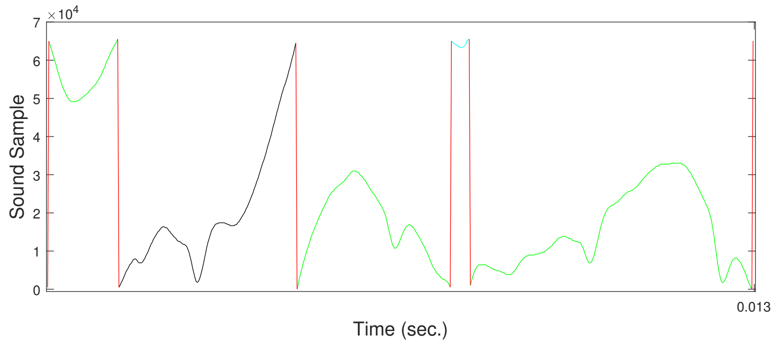
- **Vertical**: A vertical shape is similar to the slant shape, however its *length* is 1. This shape appears very often, always between two other shapes; it can thus be viewed as separator.

- **Dome**: A dome shape comprised of a sequence of three or more points ($length \geq 2$) such that the start and end points are on the same side (top or bottom) of the associated MBB. This is defined in terms of the difference between the start and end point values, which must be less than a threshold $t2$.

- **Flat**: A flat shape is similar to the dome shape, however comprised of two or more points ($length \geq 1$), and whose *depth* (maximum difference between point values) is less than a predefined threshold $t3$.



(a) Smooth shapes



(b) Fluctuated shapes

Figure 6.2: Two example Sample-PCG time series with different shapes coloured

The *type* of a *shape* is determined by the "direction" of the shape. The possible values for the *type* variable are {up, down}. The value for the *type* variable is defined by the first two points in a sequence. If the value of the second point is greater than the first (in other words, the difference is positive), the *type* is "up". If the difference is negative, the *type* is "down". Where the first two points have the same value, a rare occurrence, this is dealt with by considering their location within the overall time series. If they both feature below the average value, a threshold $t4$, they are considered to have the *type* "up", otherwise the *type* is "down".

As already noted, each segment has one or more sub-segments. A "dome" segment has at least two sub-segments, one of *type* "up" and one of *type* "down", whilst the other three shapes have at least one-segment. There is no maximum for the number of sub-segments that a segment may include. The idea is that each sub-segment represents a distinct trend, which is followed by a different trend that will be represented by another sub-segment, which may belong to the same segment or the following segment.

Whichever the case, a sub-segment, as noted in Section 4.2, is described by a tuple of the form $\langle type, length, depth \rangle$. The possible values for the *type* variable are: {up, down, flat}. Note that the values for the *type* variable associated with a sub-segment are not the same as those associated with a segment. The value for the *type* variable is defined by all points in the sequence. If the value of all points is increasing (in other words, the difference between each two points in the sequence is greater than a threshold $t5$), the *type* is "up". If the difference is less than $t5$, the *type* is "down". Otherwise the *type* is "flat".

From the foregoing, a set of five thresholds, $\{t1, t2, t3, t4, t5\}$, used to segment point series, were identified. For ease of understanding, these are summarised below and illustrated in Figure 6.3:

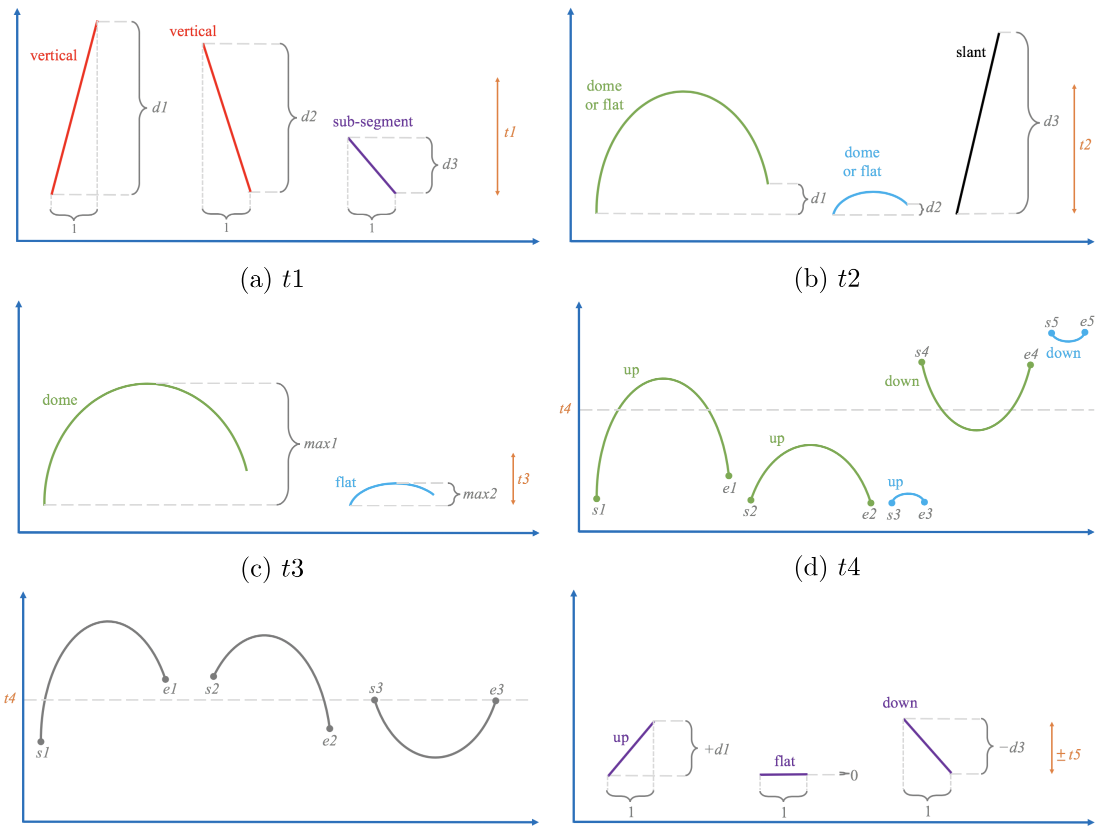
- $t1$: The "vertical" shape threshold, a value over which the difference in Sample value between two consecutive points $(p_i - p_{i-1})$ indicates a "vertical" shape; otherwise, it is a sub-segment of another shape; see Figure 6.3(a).
- $t2$: The "slant" shape threshold, a value over which the difference in Sample value between the start and end point values of a candidate segment indicates a "slant" shape, otherwise a "dome" or "flat" shape; see Figure 6.3(b).
- $t3$: The "dome" shape threshold, a value above which the maximum difference of the points of the segment in question indicates a "dome" shape, a "flat" shape otherwise, as shown in Figure 6.3(c).
- $t4$: The "dome" and "flat" shape *type* threshold, a value above which the start and end point values of a shape specifies the *type* to be "down", or "up" otherwise, see Figure 6.3(d). The value for $t4$ needs to be chosen carefully so as to prevent the special cases shown in Figure 6.3(e).
- $t5$: The sub-segment *type* threshold, a value which indicates whether the difference between any two points of a sub-segment specifies the *type* "up" $(p_i - p_{i-j} < t5)$, "down" $(p_i - p_{i-j} > t5)$ or "flat" $(p_i - p_{i-j} = t5)$, see Figure 6.3(f).

The motivations for the proposed PCG$_{\text{seg}}$ mechanism was to represent time series, and Sample-PCG point series in particular, in terms of their constituent shapes and sub-shapes in a two-level hierarchy instead of using one of the proposed techniques in the literature, such as averaging values taken periodically [96] or extracting trends [153]. The advantages of the mechanism over these existing techniques are as follows:

- The main information, which would be lost in the case of methods that generate average values, is preserved because the change rate in the Sample-PCG point series tends to be high, unless the chosen averaging period is very small (in other words, a very narrow window), which would obviate the benefit of applying the segmentation concept to the time series to reduce the length of the time series data.

- A more expressive representation can be achieved, than that produced by earlier segmentation mechanisms, by considering “parent” and “child” shapes (segments and sub-segments) where the child level represents “trends” in the parent level. Recall that a parent shape can have any number of sub-shapes (trends); this will be the case where the parent shape features many irregularities and fluctuations.

- Motif discovery using point series requires a substantial amount of matching of point series sub-sequences. The proposed two-level hierarchical segmentation allows for “early abandonment” where the parent segment does not fit the comparator segment (no need to go down to the next level). Although, for the proposed approach there is one exception (this is discussed further in Section 6.4).



(a) $t1$ (b) $t2$ (c) $t3$ (d) $t4$ (e) $t4$ exclusion (f) $t5$

Figure 6.3: Sequence of plots showing how the thresholds $t1$, $t2$, $t3$, $t4$ and $t5$ distinguish between *shapes* and *types*, and detect sub-segments

6.3.2 The PCG$_{\text{seg}}$ Algorithm

In this section, the PCG$_{\text{seg}}$ algorithm is presented and discussed. The block diagram shown in Figure 6.4 indicates the basic operations of the proposed algorithm; the directed arcs indicate the flow of control. The input is a Sample-PCG time series comprised of a sequence of points each with an associated Sample value. In the figure, i is a point index. The process proceeds by looping through the time series. From the figure, it can be seen that the algorithm first tries to determine the presence of a "vertical" segment (*shape*) because it is: (i) the simplest to identify, (ii) the most frequently occurring and (iii) comprises one sub-segment. If a "vertical" segment is found, the segment is registered. Otherwise, the process continues as in the figure until all segments in the point series have been detected.

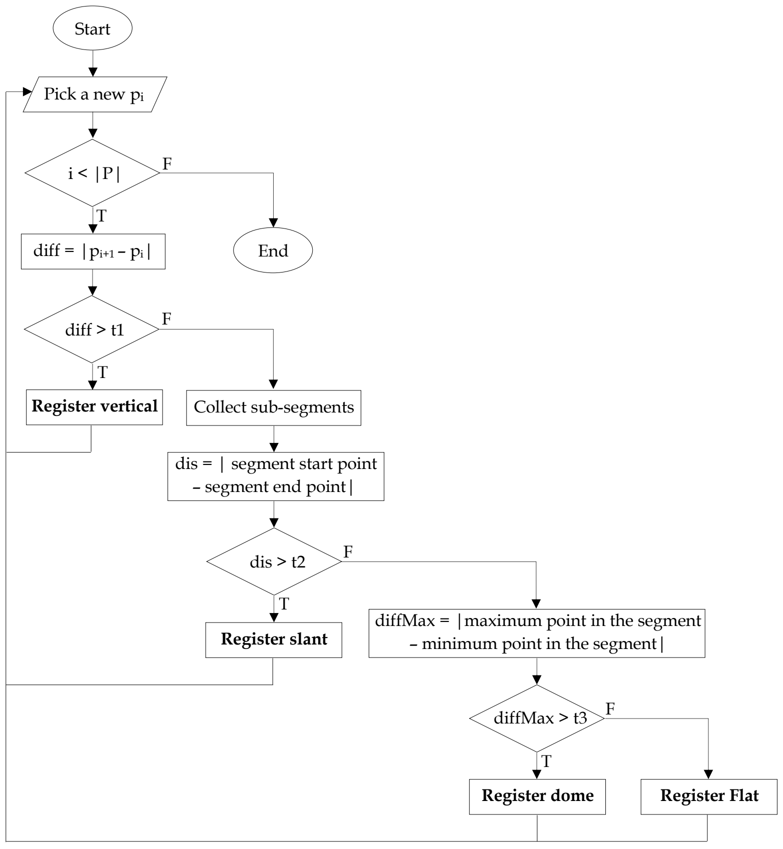


Figure 6.4: The basic operations of PCG$_{seg}$ algorithm

The pseudo code for the segmentation mechanism is given in Algorithms 4 to 8. The top-level process is given in Algorithm 4. Algorithms 5 to 7 are what are referred to as "registration" algorithms for the "vertical", the "slant", and the "dome" and the "flat" *shape*s. Algorithm 8 is the segment collection algorithm. Returning to the top-level algorithm, Algorithm 4, the inputs are: (i) a point series P comprised of a sequence of PCG Sample values $\{p_1, p_2, \dots\}$ and (ii) a set of threshold values $\{t1, t2, t3, t4, t5\}$. Note also that there are four global variables: (i) an index counter i for the point series P that is updated as the algorithm progresses, (ii) the maximum Sample value for the current segment $maxSam$, (iii) the minimum Sample value for the current segment $minSam$ and (iv) a segmented point series $\overline{P}$ comprised of a sequence of segments that are added as the algorithm progresses.

Algorithm 4 PCG$_{\text{seg}}$

```
Require: $P$, $\{t1, t2, t3, t4, t5\}$
Ensure: $\overline{P}$
 1: $i \leftarrow 1$, A global variable, an index counter for the point series $P$
 2: $\overline{P} \leftarrow \emptyset$, A global variable, a set to hold the segmented point series $P$
 3: $maxSam$, $minSam$, Global variables to hold the maximum/minimum point values of
    a segment
 4: $segStart$, A variable to hold the index value of the beginning of a segment
 5: $diff$, A variable to hold the difference between two consecutive points
 6: $dis$, A variable to hold the distance between the start and end point values of a segment
 7: while $i < |P|$ do
 8:   $diff \leftarrow p_{i+1} - p_i$
 9:   if $|diff| > t1$ then
10:     $\overline{P} \leftarrow \overline{P} \cup registerVertical(diff)$
11:     $i \leftarrow i + 1$
12:   else
13:     $segStart \leftarrow i$
14:     $maxSam \leftarrow p_i$
15:     $minSam \leftarrow p_i$
16:     $S_C \leftarrow collectSubSegments(P, diff, t1, t5)$
17:     $dis \leftarrow |p_i - p_{segStart}|$
18:     if $dis > t2$ then
19:       $\overline{P} \leftarrow \overline{P} \cup registerSlant(diff, i - segStart, S_C)$
20:     else
21:       $\overline{P} \leftarrow \overline{P} \cup registerDomeFlat(segStart, t3, t4, S_C)$
22:     end if
23:   end if
24: end while
25: return( $\overline{P}$ )
```

The algorithm begins by defining seven variables, the four global variables mentioned above (i, $\overline{P}$, $maxSam$ and $minSam$) and the following three variables: (i) $segStart$ to hold the index value of the start point for a segment, (ii) $diff$ to hold the difference between two consecutive points and (iii) dis to hold the distance between the start and end point values of a segment. The algorithm features a loop (lines 7 to 24) whereby the input point series is stepped through point-by-point using the global index i. The algorithm commences by calculating the difference ($diff$) in value between the current point and the following point (line 8). If this difference exceeds threshold $t1$, the two points represent a "vertical" shape, and a vertical segment is appended to $\overline{P}$ (line 10). Otherwise, it is a more complicated shape ("slant", "dome" or "flat") which comprises a *length* greater than one. The algorithm proceeds as follows. The start point for the segment, $segStart$ is set to the current index i (line 13), and the maximum and minimum Samples ($maxSam$ and $minSam$) found so far are set to the value of point p_i (lines 14 and 15). The set of sub-segments, S_C, within the current segment is then determined through a call to the function $collectSubSegments$ (line 16). During this process, the global point series index i

will be incremented and the two global variables $maxSam$ and $minSam$ will be updated. If the difference between the start and end points of the current segment is greater than $t2$ (line 18), it is a "slant" shape, and the segment is appended to $\overline{P}$ (line 19). Otherwise, it is either a "dome" or "flat" shape, which is resolved through a call to the function $registerDomeFlat$ (line 21). The loop then repeats until the entire point series P has been processed; $\overline{P}$ is returned on completion (line 25).

The pseudo code for the $registerVertical$ function is given in Algorithm 5. With reference to Algorithm 5, it should be noted that "vertical" segments are discovered as soon as possible in order to reduce the number of computations. However, some "vertical" shapes will not be discovered at this stage because the difference in values does not exceed the predefined threshold $t1$. They will instead be identified later in the process using the $registerSlant$ function. The input is the value difference, $diff$, between the two consecutive points making up the "vertical" shape. The output is a segment S defined in the form of a tuple, $\langle S_p, S_C \rangle$, where S_p is the parent segment and S_C is the only sub-segment. The pseudo code is self explanatory.

Algorithm 5 function registerVertical

```
Require: $diff$
Ensure: $S$
 1: $segType$, $subSegType$, Variables to hold a segment/sub-segment $type$
 2: if $diff > 0$ then
 3:     $segType \leftarrow$ "up"
 4:     $subSegType \leftarrow$ "up"
 5: else
 6:     $segType \leftarrow$ "down"
 7:     $subSegType \leftarrow$ "down"
 8: end if
 9: $S_p \leftarrow \langle$"vertical", $segType$, 1$\rangle$
10: $S_C \leftarrow \langle subSegType,\ 1,\ |diff| \rangle$
11: $S \leftarrow \langle S_p, S_C \rangle$
12: return( $S$ )
```

The pseudo code for the $registerSlant$ function is given in Algorithm 6. The algorithm takes as input: (i) the difference, $diff$, between the first two consecutive points in the current segment; (ii) the length of the segment $length$; and (iii) a set of sub-segments $S_C = \{S_{c_1}, S_{c_2}, \ldots\}$, within the current segment. As before, the output is a segment S defined in the form of a tuple, $\langle S_p, S_C \rangle$.

The pseudo code for the $registerDomeFlat$ function is given in Algorithm 7. The inputs are: (i) the segment start index ($segStart$), (ii) the thresholds $t3$ and $t4$ and (iii) the set of sub-segments, S_C, within the current segment. The algorithm also uses the global variables: (i) $maxSam$ and $minSam$, the maximum and minimum point values for the segment, and (ii) the index counter for the point series i. The output, as before, is a segment S defined in the form of a tuple, $\langle S_p, S_C \rangle$.

Algorithm 6 function registerSlant

```
Require: $diff$, $length$, $S_C$
Ensure: $S$
 1: $segType$, A variable to hold a segment $type$
 2: if $length = 1$ then
 3:     $S \leftarrow registerVertical(diff)$
 4: else
 5:     if $diff > 0$ then
 6:         $segType \leftarrow$ "up"
 7:     else
 8:         $segType \leftarrow$ "down"
 9:     end if
10:     $S_p \leftarrow \langle$"slant", $segType$, $length\rangle$
11:     $S \leftarrow \langle S_p, S_C \rangle$
12: end if
13: return( $S$ )
```

Finally, the pseudo code for the function to generate the set of one or more sub-segments that might feature in a parent segment, the *collectSubSegments* function, is given in Algorithm 8. The inputs are: (i) the given point series P, (ii) the difference, $diff$, between the first two points in the current segment and (iii) thresholds $t1$ and $t5$. The function also uses the global variable: i, $maxSam$ and $minSam$. Recall that the $maxSam$ and $minSam$ variables are used later in the process to distinguish between the "dome" and "flat" shapes. The function returns a set of one or more sub-segments $S_C = \{S_{c_1}, S_{c_2}, \dots\}$. The function loops through P from the current index identifying sub-segments until the start of a vertical line segment is found (defined by the threshold $t1$). A sub-segment ends when its *type* value changes. Recall that the values for the *type* variable are taken from the set {up, down, flat}.

In more detail, Algorithm 8 proceeds as follows. It commences (lines 1 to 3) by defining the following: (i) an empty set S_C to hold sub-segments; (ii) two variables *segStart* and *subSegStart*, and assigning the value of i to them so as to maintain the index for the start of the segment and its first sub-segment; (iii) a dynamic variable *newType* to hold the *type* of the two consecutive points under consideration in order to detect any change in the trend, hence a new sub-segment; and (iv) a variable $sofarType$ to hold the previously identified *type* so that this can be used for comparison purposes. The function then (lines 5 to 17) determines the *type* represented by the current point (p_i) and the next point (p_{i+1}) and updates the $maxSam$ and $minSam$ values as appropriate. If a new *type* is identified (line 18), this means the end of a sub-segment is reached; thus it is stored in S_C. Otherwise, the next two points are processed and compared (line 27). The algorithm stops when the end of the point series is reached (line 24) or a vertical line segment is found (line 28), the last sub-segment is then stored (line 29) and the set of sub-segments, S_C, is returned.

Algorithm 7 function registerDomeFlat

```
Require: $segStart$, $t3$, $t4$, $S_C$
Ensure: $S$
 1: $segEnd \leftarrow i$, A variable to hold the index value of the end of a segment
 2: $segShape$, $segType$, $segLength$, Variables to hold a segment shape, type and length
 3: $diffStartMax$, $diffStartMin$, $diffEndMax$, $diffEndMin$, Variables to hold the
    maximum/minimum difference of start/end point values of a segment
 4: if $segStart < t4$ and $segEnd < t4$ then
 5:    $segType \leftarrow$ "up"
 6:    $diffStartMax \leftarrow |p_{segStart} - maxSam|$
 7:    $diffEndMax \leftarrow |p_{segEnd} - maxSam|$
 8:    if $diffStartMax > t3$ or $diffEndMax > t3$ then
 9:       $segShape \leftarrow$ "dome"
10:    else
11:       $segShape \leftarrow$ "flat"
12:    end if
13: else
14:    $segType \leftarrow$ "down"
15:    $diffStartMin \leftarrow |p_{segStart} - minSam|$
16:    $diffEndMin \leftarrow |p_{segEnd} - minSam|$
17:    if $diffStartMin > t3$ or $diffEndMin > t3$ then
18:       $segShape \leftarrow$ "dome"
19:    else
20:       $segShape \leftarrow$ "flat"
21:    end if
22: end if
23: $segLength \leftarrow segEnd - segStart$
24: $S_p \leftarrow \langle segShape, segType, segLength \rangle$
25: $S \leftarrow \langle S_p, S_C \rangle$
26: return( $S$ )
```

6.3.3 Illustration

An illustration of the operation of the PCG$_{\text{seg}}$ algorithm is presented in this sub-section. Figure 6.5 shows a fragment of a PCG time series that has been segmented, into segments and sub-segments, using the proposed PCG$_{\text{seg}}$ algorithm. The individual segments are colour-coded according to *shape*, and the sub-segment borders are indicated; the borders of a sub-segment are the end point of the previous sub-segment and the start point of the following sub-segment.

The example given in Figure 6.5 comprises a sequence of 11 segments and 15 sub-segments covering 140 individual points. It can be seen that the segmented representation is much more compact compared to the Sample-PCG point series representation. The first segment in the figure is of shape "vertical" and consists of one sub-segment ($S_p = \langle$vertical, down, 1$\rangle$ and $S_{c_1} = \langle$down, 1, 61,951$\rangle$). The next segment is of shape "dome" and consists of three sub-segments ($S_p = \langle$dome, up, 54$\rangle$, $S_{c_1} = \langle$up, 25, 47,872$\rangle$, $S_{c_2} = \langle$flat, 2, 0$\rangle$ and $S_{c_3} = \langle$down, 27, 49,920$\rangle$). The "slant" shapes that appear in the figure are all

Algorithm 8 function collectSubSegments

```
Require: $P$, $diff$, $t1$, $t5$
Ensure: $S_C$
 1: $S_C \leftarrow \emptyset$, A set to hold one or more sub-segments
 2: $segStart \leftarrow i$, $subSegStart \leftarrow i$, Variables to hold the index value of the beginning of
    a segment/sub-segment
 3: $newType$, $sofarType \leftarrow null$, Variables to hold a sub-segment $type$
 4: repeat
 5:   if $diff > t5$ then
 6:     $newType \leftarrow$ “up”
 7:     if $p_{i+1} > maxSam$ then
 8:       $maxSam \leftarrow p_{i+1}$
 9:     end if
10:   else if $diff < t5$ then
11:     $newType \leftarrow$ “down”
12:     if $p_{i+1} < minSam$ then
13:       $minSam \leftarrow p_{i+1}$
14:     end if
15:   else
16:     $newType \leftarrow$ “flat”
17:   end if
18:   if $newType \neq sofarType$ and $sofarType \neq null$ then
19:     $S_C \leftarrow S_C \cup \langle sofarType, i - subSegStart, |p_{subSegStart} - p_{subSegEnd}|\rangle$
20:     $subSegStart \leftarrow i$
21:   end if
22:   $sofarType \leftarrow newType$
23:   $i \leftarrow i + 1$
24:   if $i \geq |P|$ then
25:     exit this loop
26:   end if
27:   $diff \leftarrow p_{i+1} - p_i$
28: until $|diff| > t1$
29: $S_C \leftarrow S_C \cup \langle sofarType, i - subSegStart - 1, |p_{subSegStart} - p_{subSegEnd}|\rangle$
30: return( $S_C$ )
```

similar, they are “down” slant shapes and consist of one sub-segment; the first of these is $S_p = \langle$slant, down, 17$\rangle$ and $S_{c_1} = \langle$down, 17, 62,208$\rangle$. The only “flat” shape that appears in the example consists of three sub-segments ($S_p = \langle$flat, down, 13$\rangle$, $S_{c_1} = \langle$down, 5, 1,536$\rangle$, $S_{c_2} = \langle$flat, 2, 0$\rangle$ and $S_{c_3} = \langle$up, 6, 1,536$\rangle$).

6.4 Motif Detection

In Section 6.2, a high-level overview of the proposed PCG$_{\text{seg}}$ Classification approach was presented. In the previous section, Section 6.3, the operation of the segmentation algorithm, PCG$_{\text{seg}}$, was illustrated. Once the entire point series has been segmented, the next step is the discovery (mining) of motifs. The adopted approach is based on the MK motif

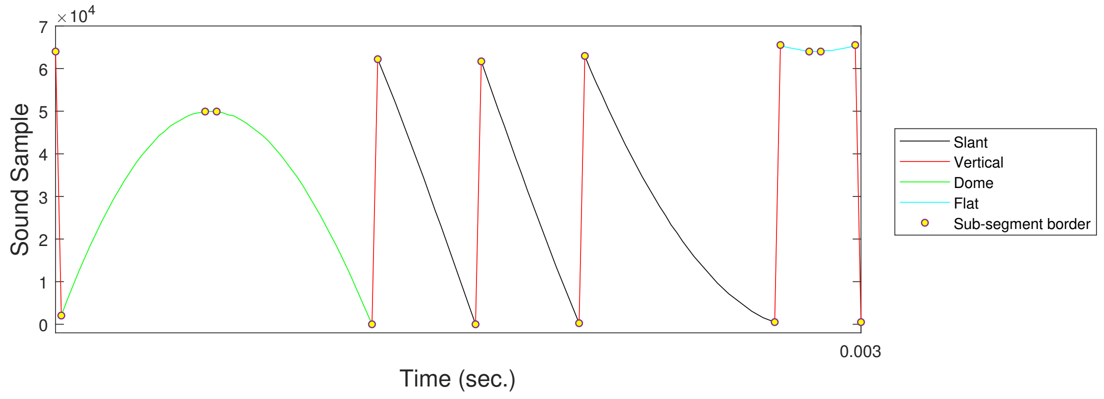


Figure 6.5: Example segmentation of part of a PCG point series with the individual segments colour-coded and the sub-segments indicated by border points

discovery algorithm [112] presented with respect to the Benchmark approach in the previous chapter, Chapter 5. As noted previously, the MK algorithm is a well-established motif detection algorithm that operates by identifying and testing a number of candidate motifs and then selecting the best ϵ among these motifs where matches are found; the recommended value for ϵ is two [112].

The original MK algorithm, as demonstrated in the previous chapter, was designed to operate with point series and not with segmented data; in other words, the original MK algorithm will not operate with a two-level hierarchical segmentation. More specifically, the similarity comparison, required in both the training and classification stages, needs to be conducted between segments, and not points. Therefore, an alternative "similarity function" was required so that distances between segments could be derived. A fairly strict similarity measure was required for motif generation. In preliminary experiments exploring the use of alternative similarity mechanisms, it was found that when using a strict similarity measure for NNC, the vast majority of previously unseen cases, 74%, were not classified at all because of the strictness of the matching. Hence, a more tolerant similarity matching for NNC was required. Therefore, two methods for calculating the similarity are proposed in this chapter:

1. **Strict Similarity Measurement**, using five criteria, for motif generation; and
2. **Tolerant Similarity Measurement** for classification purposes.

The first is discussed in Sub-section 6.4.1, and the second in Sub-section 6.4.2.

6.4.1 Strict Similarity Measurement

The Strict Similarity Measurement mechanism, as noted above, was designed for motif generation purposes using hierarchically segmented data. For the similarity measurement, five criteria were considered:

1. Number of (parent) segments.
2. Segment shapes and types.
3. Number of (child) sub-segments.
4. Sub-segment types.
5. Average sub-segment lengths and depths.

These were considered in turn in such a way that “early abandonment” could be adopted. Thus, when comparing two sub-sequences, if any of the first four criteria is not satisfied, the comparison can be stopped without further computation because the two sub-sequences under consideration will clearly not be similar. In other words, the motif discovery algorithm will consider these two sub-sequences not to be similar and start another similarity measurement using the five criteria; otherwise, a numeric value for the similarity will be returned, the closer this value is to zero the more similar the sub-sequences. A value of zero will indicate two identical sub-sequences.

In more detail, both time series sub-sequences must feature the same number of segments. The two sets of parent segments must feature the same sequence of shapes with the exception that “dome” and “flat” shapes can be matched provided they have the same type. This exception is applied because inspection of the PCG data indicated that some “flat” segments were very similar to “dome” segments and often have the same sub-segments. For two motifs to be similar, the number of sub-segments that feature in each segment should also be identical, as should the order and types of the sub-segments. Finally, if the previous criteria are satisfied, the accumulated Mean Root Square Distance (RMSD) of the lengths and depths of the sub-segments is calculated. From this, a similarity index, sim, is derived using Equation 6.1 where X and Y are the sub-sequences (candidate motifs) to be compared and $X_C = \{X_{C_1}, X_{C_2}, \ldots, X_{C_\omega}\}$ and $Y_C = \{Y_{C_1}, Y_{C_2}, \ldots, Y_{C_\omega}\}$ are sets of sub-segments held within X and Y. Recall that ω is the window size, which in this chapter is the number of segments in a sub-sequence. Note that each element (X_{C_i} and Y_{C_i}) in a segment represents a group of, one or more, sub-segments in a sub-sequence; each one of the two sets, X_C and Y_C, could be elongated to include every single sub-segment, hence, $X_C = \{X_{c_1}, X_{c_2}, \ldots, X_{c_j}\}$ and $Y_C = \{Y_{c_1}, Y_{c_2}, \ldots, Y_{c_j}\}$, where j is the total number of all sub-segments ($j \geq \omega$); j will be equivalent to ω if each segment only has one sub-segment.

$$sim = \frac{\sqrt{\sum_{i=1}^{j} (length_{X_{c_i}} - length_{Y_{c_i}})^2} + \sqrt{\sum_{i=1}^{j} (depth_{X_{c_i}} - depth_{Y_{c_i}})^2}}{j} \tag{6.1}$$

6.4.2 Tolerant Similarity Measurement

As noted earlier, the Strict Similarity Measurement mechanism with five criteria described in the previous sub-section is appropriate for motif generation, but unsuitable for measuring the similarity between motifs representing previously unseen records and motifs held in an NNC bank because the nature of the mechanism is such that it is unlikely to find a match. Therefore, an alternative, more tolerant, approach was required for the classification stage where a similarity index, sim, is calculated between a motif m_1 belonging to a previously unseen record and a motif m_2 belonging to a record in an NNC bank. The similarity index produced is a numeric index, the closer to zero the greater the similarity. The class-label associated with the most similar motif in the bank is assigned to the previously unseen record according to this similarity index. The calculation of the Tolerant Similarity Measurement was much more complex than the strict similarity measure. It can best be described in terms of the following processes:

1. The top-level process, the Tolerant Similarity function (Algorithm 9).
2. The Segment Similarity function, *segSimilarity* (Algorithm 10).
3. The Segment Similarity Zero function, *segSimilarityZero* (Algorithm 11).
4. The Sub-segment Similarity function, *subSegSimilarity* (Algorithm 12).
5. The Sub-segment Similarity Zero function, *subSegSimilarityZero* (Algorithm 13).

Each is discussed in detail in the rest of this sub-section.

The pseudo code for the top-level Tolerant Similarity Measurement process is given in Algorithm 9. The inputs are two motifs m_1 and m_2, one from a previously unseen record that we wish to classify and one from the NNC bank, each comprised a sequence of ω segments. The output is a similarity index, sim, for the two motifs. The algorithm starts by defining a number of variables: (i) two sets X and Y to hold segments from m_1 and m_2, respectively, (ii) a set $procY$ to hold segments from m_2 that have been processed so far, (iii) a set $procY'$ to hold segments from m_2 that have not been processed so far, (iv) a variable sim to hold the overall accumulated similarity index between m_1 and m_2, (v) two variables sim_{len} and sim_{dep} to hold the individual similarity values for the lengths and depths parameters of the sub-segments in m_1 and m_2 respectively, (vi) two variables $segSim_{len}$ and $segSim_{dep}$ to hold temporary similarity values for the length and depth segment parameters, respectively, for segments with the same type, (vii) two variables len and dep to hold similarity values for the *length* and *depth* sub-segment parameters, respectively, and (viii) a counter $segCount$ to hold the number of processed segments so far.

The algorithm commences by excluding all "vertical" segments from the two motifs m_1 and m_2; the segments in m_1 and m_2 are then stored in X and Y (lines 7 and 8 respectively). Each segment in X is then processed using a loop (lines 9 to 30). The similarity between

Algorithm 9 Tolerant Similarity Measurement

```
Require: $m_1$, $m_2$
Ensure: $sim$
 1: $X$, $Y$, Sets to hold segments
 2: $procY \leftarrow \emptyset$, $procY'$, Sets for insuring that all segments in $Y$ are processed
 3: $sim$, $sim_{len} \leftarrow 0$, $sim_{dep} \leftarrow 0$, Variables to hold similarity values
 4: $segSim_{len}$, $segSim_{dep}$, Variables to hold temporary similarity values for a segment
 5: $len$, $dep$, Variables to hold $length$ and $depth$ for a sub-segment
 6: $segCount$, A counter for the number of processed segments
 7: $X \leftarrow$ The set of segments in $m_1$ except "vertical" segments
 8: $Y \leftarrow$ The set of segments in $m_2$ except "vertical" segments
 9: for $i = 1$ to $|X|$ do
10:    $segSim_{len} \leftarrow 0$
11:    $segSim_{dep} \leftarrow 0$
12:    $segCount \leftarrow 0$
13:    for $j = 1$ to $|Y|$ do
14:       if $X_{i_p}.shape = Y_{j_p}.shape$ and $X_{i_p}.type = Y_{j_p}.type$ then
15:          $len, dep \leftarrow segSimilarity(X_i, Y_j)$
16:          $segSim_{len} \leftarrow segSim_{len} + len$
17:          $segSim_{dep} \leftarrow segSim_{dep} + dep$
18:          $segCount \leftarrow segCount + 1$
19:          $procY \leftarrow procY \cup Y_j$
20:       end if
21:    end for
22:    if $segCount \neq 0$ then
23:       $sim_{len} \leftarrow sim_{len} + (segSim_{len} \div segCount)$
24:       $sim_{dep} \leftarrow sim_{dep} + (segSim_{dep} \div segCount)$
25:    else
26:       $len, dep \leftarrow segSimilarityZero(X_i)$
27:       $sim_{len} \leftarrow sim_{len} + len$
28:       $sim_{dep} \leftarrow sim_{dep} + dep$
29:    end if
30: end for
31: $procY' \leftarrow Y - procY$
32: for $z = 1$ to $|procY'|$ do
33:    $len, dep \leftarrow segSimilarityZero(procY'_z)$
34:    $sim_{len} \leftarrow sim_{len} + len$
35:    $sim_{dep} \leftarrow sim_{dep} + dep$
36: end for
37: $sim \leftarrow (\sqrt{sim_{len}} + \sqrt{sim_{dep}}) \div (|X| + |procY'|)$
38: return( $sim$ )
```

each segment $X_i \in X$, and all segments of the same parent *shape* and *type* in Y (lines 13 to 21) is calculated, through a call to the *segSimilarity* function described later, and then averaged (lines 23 and 24). If the segment X_i does not have a similar segment in Y, the similarity is calculated with zeros through a call to the function *segSimilarityZero* function (line 26). Next, the algorithm compares the segments in Y with the corresponding segments in X (lines 31 to 36). Any segment in Y that does not have a corresponding segment in X is compared to a zero value (line 33) to compute its similarity. In the end, the mean root is calculated producing a similarity index, *sim* (line 37).

The pseudo code for the *segSimilarity* function, called from line 15 in the pseudo code for the top-level function, is given in Algorithm 10. The inputs are two segments $S1 = \{\langle S1_p, S1_C \rangle\}$ and $S2 = \{\langle S2_p, S2_C \rangle\}$. The outputs are the similarity index for the length and depth parameters, *len* and *dep* respectively. The algorithm commences by defining some variables: (i) two sets A and B to hold $S1_C$ and $S2_C$, where A is dynamically associated with the set where its sub-segment under consideration has the highest *length*, (ii) two variables *len* and *dep* to hold the *length* and *depth* similarity values between $S1$ and $S2$ respectively, (iii) a set of counter variables, i, j, x, y and z, (iv) two variables len_s and dep_s to hold the sub-segment *length* and *depth* similarity values when calculated, (v) two variables $lenS1$ and $lenS2$ to hold the *length* of a sub-segment under consideration in $S1_C$ and $S2_C$ respectively, and (vi) three variables $count_{cor}$, $length_{cor}$ and $nextLen$ used to synchronise a long sub-segment and the corresponding sub-segments.

The *segSimilarity* function commences by determining the number of sub-segments in both $S1$ and $S2$, the length of the sets $S1_C$ and $S2_C$ associated with $S1$ and $S2$ respectively. If the number of sub-segments is equal (lines 4 to 12), each pair of corresponding sub-segments will be processed in turn through a call to the function *subSegSimilarity*. On each iteration, the similarity for both the *length* and *depth* of the sub-segment pair will be calculated and returned by the function *subSegSimilarity*. If the number of sub-segments is not equal, the similarity between them is measured through calls to the *subSegSimilarity* function (lines 13 to 41). In the pseudo code, the variable x is the index into $S1_C$, and the variable y is the index into $S2_C$. The sub-segments lengths are compared in line 15. The sub-segments associated with the shortest *length* are stored in the variable A with index i, and the other in B with index j. For synchronisation, the function loops through A from i collecting more sub-segments in A until their cumulative length ($length_{cor}$) gets as close as possible to the *length* of the longest sub-segment (line 27). The next step is finding the similarity between the long sub-segment in B and the identified sub-segments in A, this is calculated through a call to the function *subSegSimilarity* (line 29). The indices x and y are then updated (lines 36 to 40).

Given sets of sub-segments of unequal length, once the above process has completed, there might be some sub-segments remaining in either $S1_C$ or $S2_C$. The similarity values for these remaining sub-segments are calculated assuming a zero value in the comparator sub-segment through a call to the function *subSegSimilarityZero* (lines 42 to 46 or lines 47 to 51 as appropriate).

Algorithm 10 function segSimilarity

```
Require: $S1$, $S2$
Ensure: $len$ , $dep$
 1: $A$, $B$, Sets to hold sub-segments
 2: $len \leftarrow 0$, $dep \leftarrow 0$, Variables to hold similarity values
 3: $i$, $j$, $x$, $y$, $z$, $len_s$, $dep_s$, $lenS1$, $lenS2$, $count_{cor}$, $length_{cor}$, $nextLen$, A set of used variables
 4: if $|S1_C| = |S2_C|$ then
 5:     $z \leftarrow 1$
 6:     while $z \leq |S1_C|$ do
 7:         $z \leftarrow z + 1$
 8:         $len_s, dep_s \leftarrow subSegSimilarity(S1_C, z, 1, S2_C, z, 1)$
 9:         $len \leftarrow len + len_s$ , $dep \leftarrow dep + dep_s$
10:     end while
11:     return( $len$, $dep$ )
12: end if
13: $x \leftarrow 1$ , $y \leftarrow 1$
14: repeat
15:     $lenS1 \leftarrow S1_{c_x}.length$ , $lenS2 \leftarrow S2_{c_y}.length$
16:     if $lenS1 \leq lenS2$ then
17:         $A \leftarrow S1_C$ , $B \leftarrow S2_C$ , $i \leftarrow x$ , $j \leftarrow y$
18:     else
19:         $A \leftarrow S2_C$ , $B \leftarrow S1_C$ , $i \leftarrow y$ , $j \leftarrow x$
20:     end if
21:     $length_{cor} \leftarrow 0$ , $count_{cor} \leftarrow 1$
22:     while $i \leq |A|$ do
23:         $length_{cor} \leftarrow length_{cor} + A_{c_i}.length$
24:         if $i \neq |A|$ then
25:             $nextLen \leftarrow A_{c_{i+1}}.length$
26:         end if
27:         if $i = |A|$ or $|B_{c_j}.length - length_{cor}| \leq |B_{c_j}.length - (length_{cor} + nextLen)|$ then
28:             $i \leftarrow i + 1$ , $j \leftarrow j + 1$
29:             $len_s, dep_s \leftarrow subSegSimilarity(A, i, count_{cor}, B, j, 1)$
30:             $len \leftarrow len + len_s$ , $dep \leftarrow dep + dep_s$
31:             exit this loop
32:         else
33:             $i \leftarrow i + 1$ , $count_{cor} \leftarrow count_{cor} + 1$
34:         end if
35:     end while
36:     if $lenS1 \leq lenS2$ then
37:         $x \leftarrow i$ , $y \leftarrow j$
38:     else
39:         $x \leftarrow j$ , $y \leftarrow i$
40:     end if
41: until $i > |A|$ or $j > |B|$
42: while $x \leq |S1_C|$ do
43:     $x \leftarrow x + 1$
44:     $len_s, dep_s \leftarrow subSegSimilarityZero(S1_C, x)$
45:     $len \leftarrow len + len_s$ , $dep \leftarrow dep + dep_s$
46: end while
47: while $y \leq |S2_C|$ do
48:     $y \leftarrow y + 1$
49:     $len_s, dep_s \leftarrow subSegSimilarityZero(S2_C, y)$
50:     $len \leftarrow len + len_s$ , $dep \leftarrow dep + dep_s$
51: end while
52: return( $len$ , $dep$ )
```

The pseudo code for the *segSimilarityZero* function, called from lines 26 and 33 in the top-level pseudo code (Algorithm 9) is given in Algorithm 11. The function calculates the similarity between the input segment $S = \{\langle S_p, S_C \rangle\}$ and zero because the *shape* and/or *type* of S does not exist in the other motif. The outputs are the similarity indices for the length, *len*, and for the depth, *dep*. The pseudo code is straightforward; the main idea of this function is calculating the similarity between each sub-segment in S and zero through a call to the function *subSegSimilarityZero* presented in further detail below.

Algorithm 11 function segSimilarityZero

```
Require: $S$
Ensure: $len$ , $dep$
 1: $len \leftarrow 0$ , $dep \leftarrow 0$, Variables to hold temporary similarity values
 2: $len_s$, $dep_s$, Variables to hold $length$ and $depth$ for a sub-segment
 3: $z \leftarrow 1$, A counter variable
 4: while $z \leq |S_C|$ do
 5:     $z \leftarrow z + 1$
 6:     $len_s, dep_s \leftarrow subSegSimilarityZero(S_C, z)$
 7:     $len \leftarrow len + len_s$
 8:     $dep \leftarrow dep + dep_s$
 9: end while
10: return( $len$ , $dep$ )
```

The pseudo code for the *subSegSimilarity* function, called from lines 8 and 29 in the *segSimilarity* function (Algorithm 10), is given in Algorithm 12. The inputs are: (i) two sets of constituent sub-segments, $S1_C = \{S1_{c_1}, S1_{c_2}, \dots\}$ and $S2_C = \{S2_{c_1}, S2_{c_2}, \dots\}$, (ii) two indices, *indexS*1 for $S1$ and *indexS*2 for $S2$, used to derive the start point of sub-segments, and (iii) two counters, *countS*1 for $S1$ and *countS*2 for $S2$, used to indicate how many sub-segments to pick from each set. The outputs are the length and depth similarity indices, len_s and dep_s.

Algorithm 12 commences by defining some variables: (i) two sets A and B to hold the sub-segments of $S1$ and $S2$, where A is dynamically associated with the set that participates with single sub-segment, (ii) two variables len_s and dep_s to hold similarity values for length and depth respectively between the two sub-segments, (iii) a set of variables, *currentType*, *currentLen*, *currentDep*, *typeA*, *lengthA*, *depthA*, *typeB*, *lengthB*, *depthB*, *lenDiffType*, *depDiffType* to hold sub-segment information for comparison purposes and (iv) three counter variables, i, j and *count*.

The function starts by defining the sets A and B (lines 5 to 17); it assigns the value of the set that participates with single sub-segment to the variable A; derives the value of its index i (so that we have the index for the sub-segment); assigns the value of the other set, which participates with multiple sub-segments, to the variable B; derives the value of its index j and assigning how many sub-segments it participates with to the variable *count*. The values of *type*, *length* and *depth* of the single sub-segment under consideration in A is easily found (lines 18 to 23). On the other hand, to calculate these values of some *count*

Algorithm 12 function subSegSimilarity

```
Require: $S1_C$, $indexS1$, $countS1$, $S2_C$, $indexS2$, $countS2$
Ensure: $len_s$ , $dep_s$
1: $A$, $B$, Sets to hold sub-segments
2: $len_s$ , $dep_s$, Variables to hold similarity values
3: $currentType$, $currentLen$, $currentDep$, $typeA$, $lengthA$, $depthA$, $typeB$, $lengthB \leftarrow 0$, $depthB \leftarrow 0$, $lenDiffType \leftarrow 0$, $depDiffType \leftarrow 0$, Variables to hold sub-segments' information
4: $i$, $j$, $count$, Counter variables
5: if $countS1 = 1$ then
6:     $A \leftarrow S1_C$
7:     $B \leftarrow S2_C$
8:     $i \leftarrow indexS1 - countS1$
9:     $j \leftarrow indexS2 - countS2$
10:    $count \leftarrow countS2$
11: else
12:    $A \leftarrow S2_C$
13:    $B \leftarrow S1_C$
14:    $i \leftarrow indexS2 - countS2$
15:    $j \leftarrow indexS1 - countS1$
16:    $count \leftarrow countS1$
17: end if
18: $typeA \leftarrow A_{c_i}.type$
19: $lengthA \leftarrow A_{c_i}.length$
20: $depthA \leftarrow A_{c_i}.depth$
21: if $typeA =$ "down" then
22:    $depthA \leftarrow depthA \times -1$
23: end if
24: for $z = 1$ to $count$ do
25:    $currentType \leftarrow B_{c_{z+j}}.type$
26:    $currentLen \leftarrow B_{c_{z+j}}.length$
27:    $currentDep \leftarrow B_{c_{z+j}}.depth$
28:    if $currentType \neq typeA$ then
29:        $lenDiffType \leftarrow lenDiffType + currentLen$
30:        $depDiffType \leftarrow depDiffType + currentDep$
31:    end if
32:    if $currentType =$ "down" then
33:        $currentDep \leftarrow currentDep \times -1$
34:    end if
35:    $lengthB \leftarrow lengthB + currentLen$
36:    $depthB \leftarrow depthB + currentDep$
37: end for
38: $len_s \leftarrow (lengthA - lengthB)^2 + (lenDiffType)^2$
39: $dep_s \leftarrow (depthA - depthB)^2 + (depDiffType)^2$
40: return( $len_s$ , $dep_s$ )
```

sub-segments in B, the function loops *count* times through B starting from j (lines 24 to 37); in each iteration, the algorithm adds the *length* and *depth* of the sub-segment to the variables $lengthB$ and $depthB$ (lines 35 and 36). However, some values reflecting the dissimilarity between the single sub-segment in A and the corresponding sub-segments in B are needed. The variables $lenDiffType$ and $depDiffType$ are therefore used (lines 28 to 31). In the case when a sub-segment in B has a *type* different than the single sub-segment in A, the *length* and *depth* values of the sub-segment with a different *type* will be assigned to $lenDiffType$ and $depDiffType$ (lines 29 and 30).

After finishing this loop, the similarity values len_s and dep_s are calculated in a similar manner as in the strict method, described above using Equation 6.1, which computes the accumulated Mean Root Square Distance over the *length*s and *depth*s of the sub-segments. It should be noted that the "mean root" is calculated in the end, line 37 in Algorithm 9.

The pseudo code for the *subSegSimilarityZero* function, called from lines 44 and 49 in Algorithm 10 and line 6 in Algorithm 11, is given in Algorithm 13. The inputs are: (i) a set, S_C, of constituent sub-segments $\{S_{c_1}, S_{c_2}, \dots\}$ and (ii) an index, $indexS$, indicates the location of the concerned sub-segment in S_C. The outputs are the similarity index of the *length*, len_s, and the similarity index of the *depth*, dep_s. The pseudo code is self-explanatory.

Algorithm 13 function subSegSimilarityZero

```
Require: $S_C$, $indexS$
Ensure: $len_s$ , $dep_s$
1: $len_s$ , $dep_s$, Variables to hold temporary similarity values
2: $lenS$, $depS$, Variables to hold $length$ and $depth$ for a sub-segment
3: $z \leftarrow indexS - 1$, A counter variable
4: $lenS \leftarrow S_{c_z}.length$
5: $depS \leftarrow S_{c_z}.depth$
6: $len_s \leftarrow (lenS)^2$
7: $dep_s \leftarrow (depS)^2$
8: return( $len_s$ , $dep_s$ )
```

6.5 Evaluation

This section considers the evaluation of the proposed PCG$_{\text{seg}}$ Classification approach. A similar set of experiments to those conducted with respect to the Benchmark approach presented in the previous chapter was undertaken using the collected Sample-PCG data set. The objectives of the evaluation were:

Objective 1, Operational Analysis: To investigate the operation of the proposed PCG$_{\text{seg}}$ Classification approach (both the pre-processing element and the classification element).

Objective 2, Most Appropriate Classifier: To identify the most appropriate classification model. The same two models as considered previously were evaluated: (i) Nearest Neighbour Classification (NNC) and (ii) Smallest Average Classification (SAC).

Objective 3, Comparison with Benchmark PCG Classification Approach: To compare the operation of the proposed PCG_{seg} Classification approach with the Benchmark approach from the previous chapter in terms of computational efficiency (runtime) and effectiveness (accuracy).

Objective 4, Most Appropriate Parameter Settings: To establish the most appropriate parameter settings for: (i) ω, the size of the candidate motifs and (ii) r, the number of MK references.

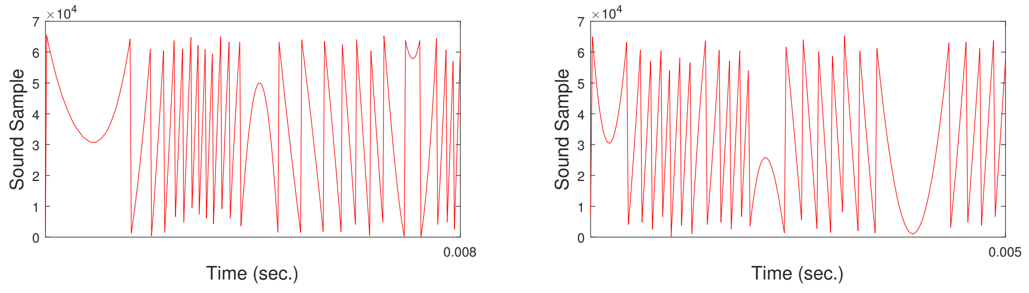
Each of these objectives is considered in turn in the following four sub-sections, Sub-sections 6.5.1 to 6.5.4. The evaluation metrics recorded were accuracy (acc), precision (prec), recall (rec), F-score (f-s) and runtime. Five-fold cross-validation was adopted and the data was statically stratified with respect to all the reported experiments. The PCG_{seg} Classification approach was implemented using the Java programming language. The experiments were run on an iMac Pro (2017) computer with 8-Cores, 3.2GHz Intel Xeon W CPU and 19MB RAM.

6.5.1 Operational Analysis

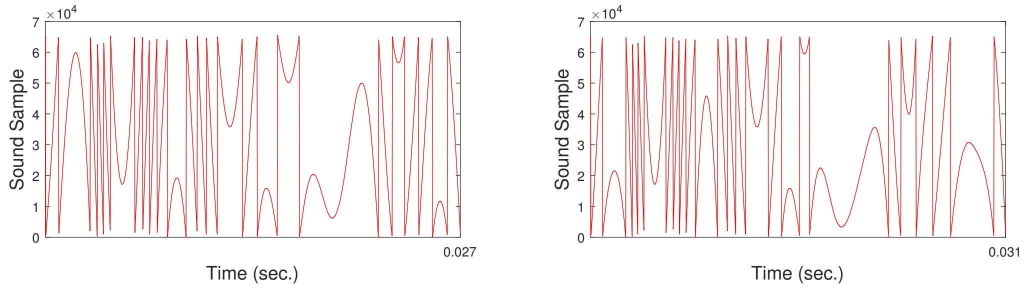
In this sub-section, the evaluation of the operation of the proposed PCG_{seg} Classification approach is presented. Examples of Sample-PCG motifs for each of the four classes obtained during the evaluation of the approach are given in Figure 6.6. These were all generated using the proposed PCG_{seg} Classification approach with $\epsilon = 2$. As the figure shows, each pair of motifs has the same number and *type* of segments and sub-segments, but with different *length*s and *depth*s.

Using NNC and five-fold cross-validation, the best and worst accuracies recorded were 72.0% and 58.3%. The precisions, recalls and F-scores were 0.181, 0.281 and 0.211 for the best accuracy, and 0.041, 0.250 and 0.065 for the worst. In addition, the recorded standard deviation for the PCG_{seg} classification model was good, giving an average of around 0.04. The average runtime to extract $\epsilon = 2$ motifs from one record was less than 12 minuets. The total time for the whole data set was roughly eleven and a half hours, on average, for each experiment; a set of nine experiments was conducted to determine the best values for the MK variables, more details concerning these experiments are given in Sub-section 6.5.4.

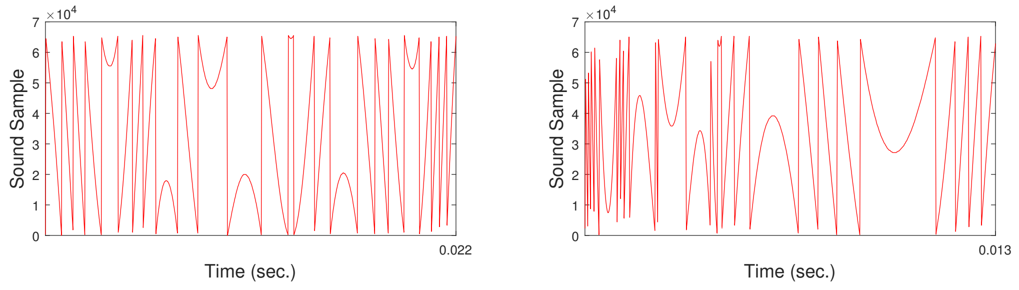
The number of data points in the point series within the evaluation data set was more than 25.5 million. Closer inspection of the operation of the PCG_{seg} Classification approach indicated that once the proposed PCG_{seg} segmentation technique had been applied, the length was reduced to some 6 million segments. A reduction in the data volume by a



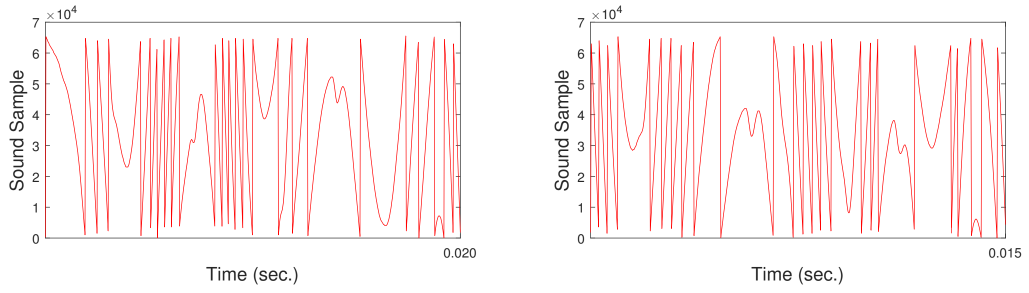
(a) Class# 1



(b) Class# 2



(c) Class# 3



(d) Class# 4

Figure 6.6: Examples of pairs of motifs, for each class in the segmented Sample-PCG evaluation data set, generated using the proposed PCG$_{seg}$ Classification approach

factor of more than four; it was anticipated that this would be reflected in the required model generation runtime as discussed later in Sub-section 6.5.3. Some further statistics related to segmentation results using the PCG$_{\text{seg}}$ approach are given in Appendix A.

6.5.2 Most Appropriate Classifier

As already noted above, and similar to the evaluation in the previous chapter, Chapter 5, two classification models were considered with respect to the evaluation presented here: NNC and SAC. In this sub-section, the performance of these models is considered in terms of accuracy, precision, recall and F-score. For NNC, $k_{nnc} = 1$ was used although, as discussed later in Sub-section 6.5.4, no performance difference was found when considering a higher value for k_{nnc} such as $k_{nnc} = 3$.

The results obtained from the two classifiers, NNC and SAC, are presented in Table 6.2. As can be seen from the table, the accuracy using NNC was better than that obtained using SAC; although there are two cases where the accuracy using NNC and SAC were the same, including the best accuracy obtained among the nine experiments. Likewise, the precision and recall results using NNC were better than those obtained using SAC. Therefore, with respect to the rest of the evaluation presented in this section, only results obtained using NNC are reported.

Table 6.2: Comparison of classification performance using NNC and SAC with respect to the proposed PCG$_{\text{seg}}$ Classification approach

ω	r	NNC ($k_{nnc} = 1$ or 3)				SAC			
		Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
25	2	**0.635**	**0.112**	**0.316**	**0.164**	0.618	0.074	0.258	0.112
	4	**0.583**	0.041	**0.250**	0.065	0.575	**0.085**	0.211	**0.105**
	6	**0.711**	**0.105**	**0.250**	**0.147**	0.660	0.104	0.243	0.139
50	2	**0.711**	**0.105**	**0.250**	**0.147**	**0.711**	**0.105**	**0.250**	**0.147**
	4	**0.661**	**0.142**	**0.203**	**0.165**	0.644	0.117	0.191	0.144
	6	**0.720**	0.181	0.281	0.211	**0.720**	**0.190**	**0.288**	**0.221**
75	2	**0.711**	**0.105**	**0.250**	**0.147**	0.591	0.088	0.181	0.114
	4	**0.670**	0.151	0.246	0.186	0.642	**0.161**	**0.293**	**0.205**
	6	**0.668**	**0.197**	**0.266**	**0.221**	0.643	0.162	0.257	0.196

6.5.3 Comparisons with Benchmark PCG Classification Approach

To evaluate the performance of the proposed segmentation approach presented in this chapter, its operation was compared with the Benchmark PCG Classification approach presented in Chapter 5, in terms of computational efficiency and effectiveness. The recorded runtime and accuracy results are presented in Tables 6.3 and 6.4 alongside the Benchmark approach runtime and accuracy results. In Table 6.3, the runtimes for the PCG$_{\text{seg}}$ Classification approach are divided into the runtime required for segmentation and the runtime required for motif discovery (model generation). From the table, it can firstly be noted that the runtime required to conduct the segmentation was negligible. From the table, it can also be clearly seen that the runtime for model generation using the MK algorithm, with $\epsilon = 2$ motifs, with respect to the PCG_{seg} Classification approach was much better than the time required with respect to the Benchmark approach. It required, on average, less than 12 minutes per record using the PCG_{seg} Classification approach, whereas the Benchmark approach required more than 2 hours per record. The proposed PCG_{seg} Classification approach therefore reduced the model generation time by a factor of more than eleven without adversely affecting the accuracy. With reference to Table 6.4, the best recorded accuracy for the PCG_{seg} Classification approach was 72.0%, whereas 71.1% was the best recorded accuracy with respect to the Benchmark approach. Note that different values were used for ω to reflect the change in data set size.

Table 6.3: Recorded runtime results (hh:mm:ss format) for the PCG_{seg} and Benchmark approaches

PCG_{seg} Approach				Benchmark Approach		
ω	r	PCG_{seg}	MK	ω	r	MK
25	2	00:00:00	00:10:02	250	2	00:42:39
	4		00:06:04		4	00:51:18
	6		00:13:51		6	00:59:25
50	2		00:09:01	500	2	00:51:11
	4		00:10:52		4	01:02:14
	6		00:07:00		6	01:26:27
75	2		00:18:03	750	2	04:10:22
	4		00:10:45		4	04:32:22
	6		00:19:08		6	05:17:44
Average		00:11:38		02:12:38		

Table 6.4: Recorded accuracy, precision, recall and F-score for the PCG_{seg} and Benchmark approaches

PCG_{seg} Approach						Benchmark Approach					
ω	r	Acc	Prec	Rec	F-s	ω	r	Acc	Prec	Rec	F-s
25	2	0.635	0.112	0.316	0.164	250	2	0.635	0.133	0.206	0.159
	4	0.583	0.041	0.250	0.065		4	0.685	0.148	0.226	0.175
	6	0.711	0.105	0.250	0.147		6	0.617	0.097	0.146	0.116
50	2	0.711	0.105	0.250	0.147	500	2	**0.711**	0.225	0.351	0.270
	4	0.661	0.142	0.203	0.165		4	0.694	0.174	0.291	0.213
	6	**0.720**	0.181	0.281	0.211		6	0.636	0.195	0.179	0.177
75	2	0.711	0.105	0.250	0.147	750	2	0.643	0.154	0.183	0.160
	4	0.670	0.151	0.246	0.186		4	0.678	0.189	0.286	0.224
	6	0.668	0.197	0.266	0.221		6	0.696	0.290	0.296	0.282

6.5.4 Most Appropriate Parameter Settings

The proposed PCG_{seg} segmentation algorithm used five thresholds: (i) $t1$ to define the "vertical" shape, (ii) $t2$ to identify the "slant" shape, (iii) $t3$ to distinguish between "dome" and "flat" shapes, (iv) $t4$ to determine the *type* of a "dome" or "flat" shapes, and (v) $t5$ to determine the *type* of a sub-segment. A set of experiments, not reported here, was conducted, using a range of values between the minimum and maximum values within the Sample-PCG data used with respect to evaluation presented in this thesis, for the first four thresholds whilst the fifth threshold was set to 0 to detect any change in the trend. The experiments suggested a range of values for each one of the four thresholds, an average of each range was used with respect to the experiments reported here, namely: $t1$ = 32,500, $t2$ = 10,000, $t3$ = 2,000 and $t4$ = 32,768. A later experiment was conducted to visualise the segmentation results which showed that the algorithm can detect all the segments and their sub-segments and distinguish between different *shape*s and *type*s in all the records in the evaluation data set. This is represented in Figure 6.5. The segmentation was also applied to some other files [1, 2, 48, 184] with the same format, these were also segmented perfectly. An example of generated segments and their sub-segments was shown in Figure 6.5.

The MK motif discovery algorithm also uses two parameters: (i) ω to define the size (length) of the candidate motifs, and (ii) r to determine the number of references in similarity calculation. Experiments were conducted using a number of values for these parameters; $\omega = \{25, 50, 75\}$, which were selected to get the best accuracies, and $r = \{2, 4, 6\}$ as used in the benchmark approach. The three values for r and three values for ω combined to give nine separate combinations; thus, nine experiments.

The criteria used for the evaluation was classification performance using NNC because earlier experiments, reported in Sub-section 6.5.2, had demonstrated that this classification model produced good results. Recall that NNC classification uses a parameter k_{nnc}, the number of votes for a class-label to be assigned. Experiments, not reported here, using $k_{nnc} = 1$ and $k_{nnc} = 3$ demonstrated that both values produced the same class-labels (as in the case of the Benchmark PCG Classification approach). Only results using $k_{nnc} = 1$ are therefore reported in the remainder of this sub-section.

The results obtained are presented in Figure 6.7 in terms of a sequence of five plots. The values for the first four plots (accuracy, precision, recall and F-score) were given earlier in Table 6.2. The best accuracy was obtained using $\omega = 50$ and $r = 6$ (72.0%). The lowest accuracy score was obtained using $\omega = 25$ and $r = 4$. It is quite arbitrary as to how the accuracy, precision and recall behave; in other words, no relation between their values, increasing or decreasing with the parameter settings, was observed.

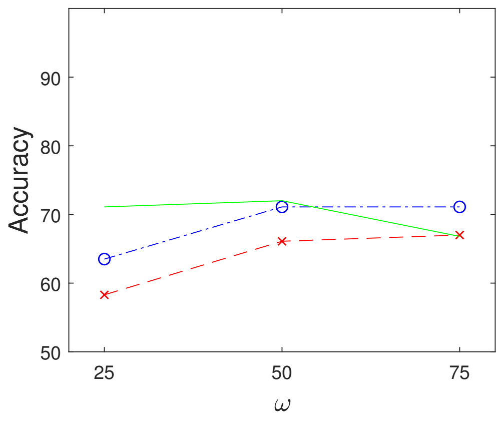
The recorded runtime results were presented previously in Table 6.3, but are shown in graph form in Figure 6.7(e) in mm:ss format. The runtime required by the proposed approach, to complete all nine experiments, was less than 4 days. It is difficult to deduce very much from the figure. From the figure, it can be seen that the shortest runtime obtained was when using $r = 4$ and $\omega = 25$; the longest when $r = 6$ and $\omega = 75$. There does not seem to be any clear correlation between runtime and the value of r and/or ω; the reason for this is discussed in further detail in the following section.

6.6 Discussion

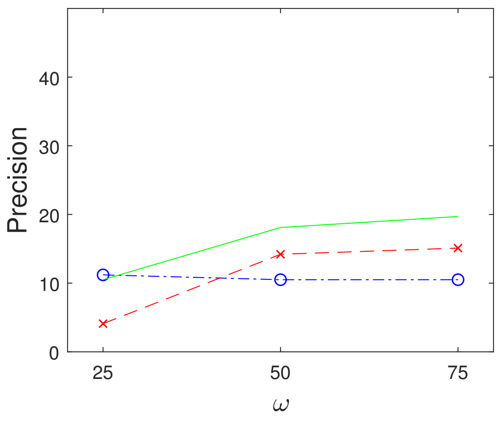
Given the results presented in the previous section, it can be concluded that, when using the PCG_{seg} Classification approach, the PCG data volume was reduced by a factor of more than four. This was reflected in the recorded runtimes, which were reduced significantly; by a factor of more than eleven. This runtime advantage is then the principal benefit offered by the proposed PCG_{seg} Classification approach, although there is also a small increase in accuracy (see Table 6.4). Using the PCG_{seg} Classification approach to extract two motifs from a PCG time series record required less than 12 minutes on average; whilst in the case of the Benchmark approach (which used unsegmented records) an average of 2 hours and 12 minutes was required to extract two motifs from a PCG record (see Chapter 5).

Looking at the results presented in Figure 6.7 in more detail, it is clear there is a great deal of variability. The variability of the results was attributed to the fact that the MK algorithm selects motifs in a random manner. Since the algorithm chooses reference sub-sequences randomly, this randomness probably does not work well with high data volumes (segmented or otherwise); the Sample-PCG time series data comprised some 6 million segments. It is conjectured that the limited performance effectiveness reported above could be as a consequence of the random manner that candidate motifs were chosen when using the MK algorithm, which may in turn have led to the selection of motifs that were not especially indicative of a class. The MK algorithm chooses the most similar sub-sequences to be motifs, whilst not taking into consideration whether these motifs are in fact good indicators of class or not. In other words, the chosen motifs may not be the best representatives of class. This might be solved by finding the most frequent motifs as these might be a better indicator of class. Furthermore, the chosen motifs in the "motif bank" from a specific class could be, as a following step, compared against each other to keep only the best representatives of class. This idea is explored in more detail later in this thesis.

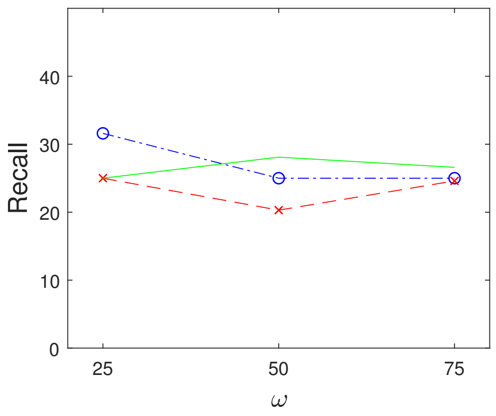
A further cause for inaccuracy is the presence of "silent gaps" in the PCG data between heartbeats, the MK algorithm does not exclude these which could easily be identified as motifs despite not being indicative of any class. Therefore, it is suggested that these silent gaps should be excluded when finding motifs. This might be done by pruning the time series so that only sub-sequences that are indicative of a class are retained. This idea is explored further later in this thesis. It is also conjectured that smoothing and filtering may contribute to classification effectiveness as it will serve to reduce runtime and remove noise. However, these last two ideas are not explored further in this thesis and, instead, are left as an item for future work.



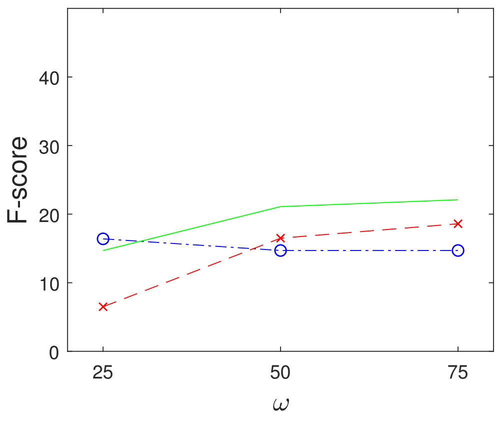
(a) Accuracy



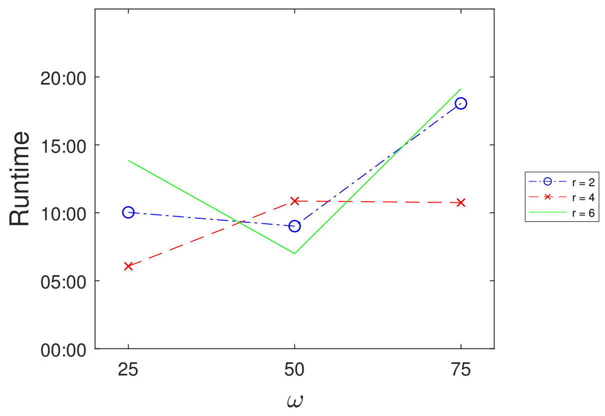
(b) Precision



(c) Recall



(d) F-score



(e) Runtime

Figure 6.7: Sequence of plots recording the results obtained from the evaluation of the proposed PCG$_{\text{seg}}$ Classification approach using Sample-PCG evaluation data and NNC

6.7 Summary

This chapter has presented the PCG$_{seg}$ Classification approach for Phonocardiogram classification using the concept of motifs. Central to the approach is a mechanism for the segmentation of Sample-PCG data using a technique based on “shapes”; more specifically, using the PCG$_{seg}$ segmentation algorithm. The objective of the work presented in this chapter was to pre-process the Sample-PCG data in order to reduce the overall size of the data, and hence reduce the execution time required for motif discovery. The evaluation results of the proposed approach, compared to the Benchmark approach presented in the previous chapter were good; runtime was reduced by a factor of more than eleven. To obtain further runtime gains, the next chapter considers the idea of removing “silent gaps” from PCG data and using frequency as a mechanism for identifying motifs. The intuition underpinning these ideas was that they would not only further reduce the motif discovery runtime, but also improve the quality of the identified motifs.

Chapter 7

The SGR-FMD Approach for Phonocardiogram Classification

7.1 Introduction

In the previous chapter, the PCG_{seg} approach to Phonocardiogram (PCG) classification was presented. This comprised a segmentation technique which was applied to the Sample-PCG time series data, and a motif discovery algorithm applied to the segmented data. The evaluation of the PCG_{seg} approach, reported in the previous chapter, indicated that the segmentation pre-processing technique reduced the overall size of the input data and hence the required runtime to identify motifs was decreased dramatically. However, the runtime was still too high. Improving the runtime is the central theme of this chapter. More specifically the central idea promoted in this chapter is an alternative pre-processing mechanism, namely to prune the time series by removing sub-sequences that are unlikely to be representative of any class-label.

From a generic time series analysis perspective, three different categories of time series sub-sequences, categorised by the author, are suitable for pruning can be identified:

1. **Common Sub-sequences**: Sub-sequences that exist in every time series and hence are not representative of any particular class.
2. **Rare Sub-sequences**: Sub-sequences that appear so infrequently so they cannot be deemed to be relevant.
3. **Poor Discriminator Sub-sequences**: Sub-sequences that appear across two or more classes and thus cannot be usefully employed to discriminate between classes.

The precise nature of the most appropriate pre-processing technique to be adopted is very much dependent on the application domain under consideration; however, the work presented in this thesis is directed at the classification of PCG signals according to a variety of heart conditions. In the context of the above categorisation, the following can be pruned from PCG time series:

1. **Common Sub-sequences**: Sub-sequences representing “silent gaps” (because they occur in every PCG time series and thus cannot be deemed to be suitable discriminative).

2. **Rare Sub-sequences**: Sub-sequences that cannot be frequent.

3. **Poor Discriminator Sub-sequences**: Discounting sub-sequences that are not good discriminators of a particular class.

With respect to the work presented in this chapter, to remove “silent gaps” from PCG time series, the used data representation was the Amplitude-PCG time series where the plots show the silent gaps in a heartbeat sequence is close to the zero value in the Amplitude axis, see Figure 2.3(a). The proposed approach to PCG time series classification in this chapter, building on earlier work, is thus referred to as the Silent Gap Removal - Frequent Motif Detection (SGR-FMD) approach, which comprises two steps:

1. **Silent Gap Removal**: A pre-processing technique that includes the removal of silent gaps; the $\mathrm{PCG_{sgr}}$ technique.

2. **Frequent Motif Detection** (FMD): A three-step algorithm that includes:

   - **Candidate Frequent Motif Generation**: Removing of sub-sequences that clearly cannot be frequent, using a novel technique involving clustering.
   - **Frequent Motif Discovery for Local Class Discrimination**: Discounting sub-sequences that are considered not to be good discriminators of a single class.
   - **Frequent Motif Selection for Global Class Discrimination**: Identifying sub-sequences that are considered to be good discriminators of all classes by insuring any good discriminator of a class, given by the previous step, does not contradict a discriminator in another class. The process of identifying good discriminators is fairly standard.

The relevant symbols with respect to this chapter are presented in Table 7.1. This includes some additional specific symbols not included in Chapter 4 where the formalism used in this thesis was first introduced.

The remainder of this chapter is organised as follows. The Silent Gap Removal - Frequent Motif Detection approach is presented in Section 7.2. Section 7.3 provides a description of the proposed $\mathrm{PCG_{sgr}}$ technique. Then, Section 7.4 presents the FMD technique. In Section 7.5, an extensive evaluation of the results obtained from experiments conducted using the proposed approach, the SGR-FMD approach, is given. Some discussion is presented in Section 7.6; and finally, in Section 7.7, some conclusions concerning the material presented in this chapter are provided.

Table 7.1: SGR-FMD approach symbol table

p	A numeric value representing a point in a point (time) series.
P	A point (time) series comprising a sequence of points $\{p_1, p_2, \dots\}$ and, in the case of PCGs, consisting of a number of cardiac cycles.
P'	A pruned point series $\{p_1, p_2, \dots\}$, $P' \subset P$.
C	A set of classes $\{c_1, c_2, \dots\}$.
ω	A "window" size used to define the length of a point series sub-sequence.
q	A point series sub-sequence of length ω, $q \subset P'$.
m	A motif, a sub-sequence q that is deemed (in some sense) to be representative of a class.
zm	The zero-motif, a hypothetical motif comprised of only zero values.
T	A set of point series and class-label pairs $\{\langle P_1, c_1\rangle, \langle P_2, c_2\rangle, \dots\}$.
Q	A set of sub-sequences of length ω, there will be a maximum of $x - \omega + 1$ sub-sequences in a points series of length x.
CL	An ordered set of clusters of sub-sequences $q_i \in Q$, $\{CL_1, CL_2, \dots\}$, ordered descendingly according to size.
Q'	The set of sub-sequences in Q that are considered frequent according to the clustering.
D_z	A set holding similarity values $\{d_{z_1}, d_{z_2}, \dots\}$, where d_{z_i} corresponds to $q_i \in Q$.
f	The frequency with which a sub-sequence q occurs in P'.
M	A set of motif and frequency value pairs $\{\langle m_1, f_1\rangle, \langle m_2, f_2\rangle, \dots\}$.
L	A set of motif and class-label pairs $\{\langle m_1, c_1\rangle, \langle m_2, c_2\rangle, \dots\}$, drawn from P'.
D	A set of motif and class-label pairs $\{\langle m_1, c_1\rangle, \langle m_2, c_2\rangle, \dots\}$, drawn from T.
D'	The set of motifs in D that are good global class-label discriminators.
D''	The set of motifs in D that are not good global class-label discriminators, $D' \cap D'' = \emptyset$.
G	A histogram of some given values.
G'	A smoothed histogram G using the parameter μ_e.
A	The set of local maxima in G'.
E	The energy of P comprising a sequence of values $\{e_1, e_2, \dots\}$.
N	The spectral centroid of P comprising a sequence of values $\{n_1, n_2, \dots\}$.
σ_m	A pre-specified frequency threshold.
λ	A pre-specified similarity threshold for comparing two sub-sequences or motifs.
θ	A dynamically computed threshold for the number of clustered sets in Q.
α	A pre-specified threshold for the number of clusters retained.
t	A dynamically computed threshold for detecting silent gaps in E or N.
μ_e	A pre-specified smoothing window parameter.
win	A pre-specified threshold for the length of non-overlapping sub-sequences of a point series P.
max	A pre-specified maximum size for a set of sub-sequences Q.
k	A pre-specified threshold limiting the number of motifs in M to the k most frequently occurring motifs, $k < max$.

7.2 The SGR-FMD Approach

This section presents the proposed Silent Gap Removal - Frequent Motif Detection PCG classification approach. A schematic of the proposed SGR-FMD approach is given in Figure 7.1. The schematic also shows the interconnection between the two individual processes that make-up the SGR-FMD approach. In the figure, and similar to the schematics given in Chapters 5 and 6, the training data set contains 9 labelled time series: 4, 3 and 2 time series belong to the classes blue, green and red respectively. It should be noted that, according to the evaluation presented later in this chapter, the number of motifs of any class retained at the end of the process (in the "motif bank") does not seem to be unduly affected by the number of time series associated with an individual class in the training data set. In other words, in the example, the blue class is associated with the highest number of time series; but at the end of the process, it has the lowest number of motifs. From the figure, it should also be noted that the multiple arrows which enter and exit the Silent Gap Removal pre-process, and the first and second sub-processes, are intended to indicate that the time series could be processed in parallel; whereas this is not the case with respect to single arrow inputs as in the case of the third sub-process which only commences when a complete set of motifs so far is available.

In more detail, the "Silent Gap Removal" mechanism, PCG_{sgr}, takes each time series (PCG) and prunes it by removing the silent gaps; the silent gaps are marked with an "**x**", while the remainder are marked with a "✓" (voiced). The pruned time series are then sent to the next process for motif generation.

During "Candidate Frequent Motif Generation", the time series sub-sequences (candidate motifs) are generated. Each time series (PCG signal) in the input is divided into sub-sequences of some predefined length ω (the window size). In this step, the distance (d) between each sub-sequence and a "zero-motif" is calculated (the zero-motif concept is discussed in further detail in Sub-section 7.4.1). After all the distance values for all sub-sequences in the time series have been calculated, they are clustered and ordered according to size. The θ largest clusters are retained; in the example given in Figure 7.1, for simplicity, θ is set to 1. The sub-sequences in that cluster are then sent to the next step.

During the "Local Frequent Motif Discovery" process, some max sub-sequences are randomly selected to measure their frequency in Q'. Note that the random selection is on the cluster level, which is already frequent, not on the whole time series level; in other words, its negative effects, if any, are very limited compared to the random effects associated with the MK algorithm discussed in the previous chapter. The k most frequent sub-sequences are selected to be the "local" motifs, which means the best motifs for the current time series P, not the entire class. A collection of k "local" motifs from each time series in the training data set is then processed in the final sub-process; as shown in the example given in Figure 7.1 where the 9 labelled time series in the training data set produced $k \times 9$ motifs.

In the final process, "Global Frequent Motif Selection", the motifs are grouped accord-

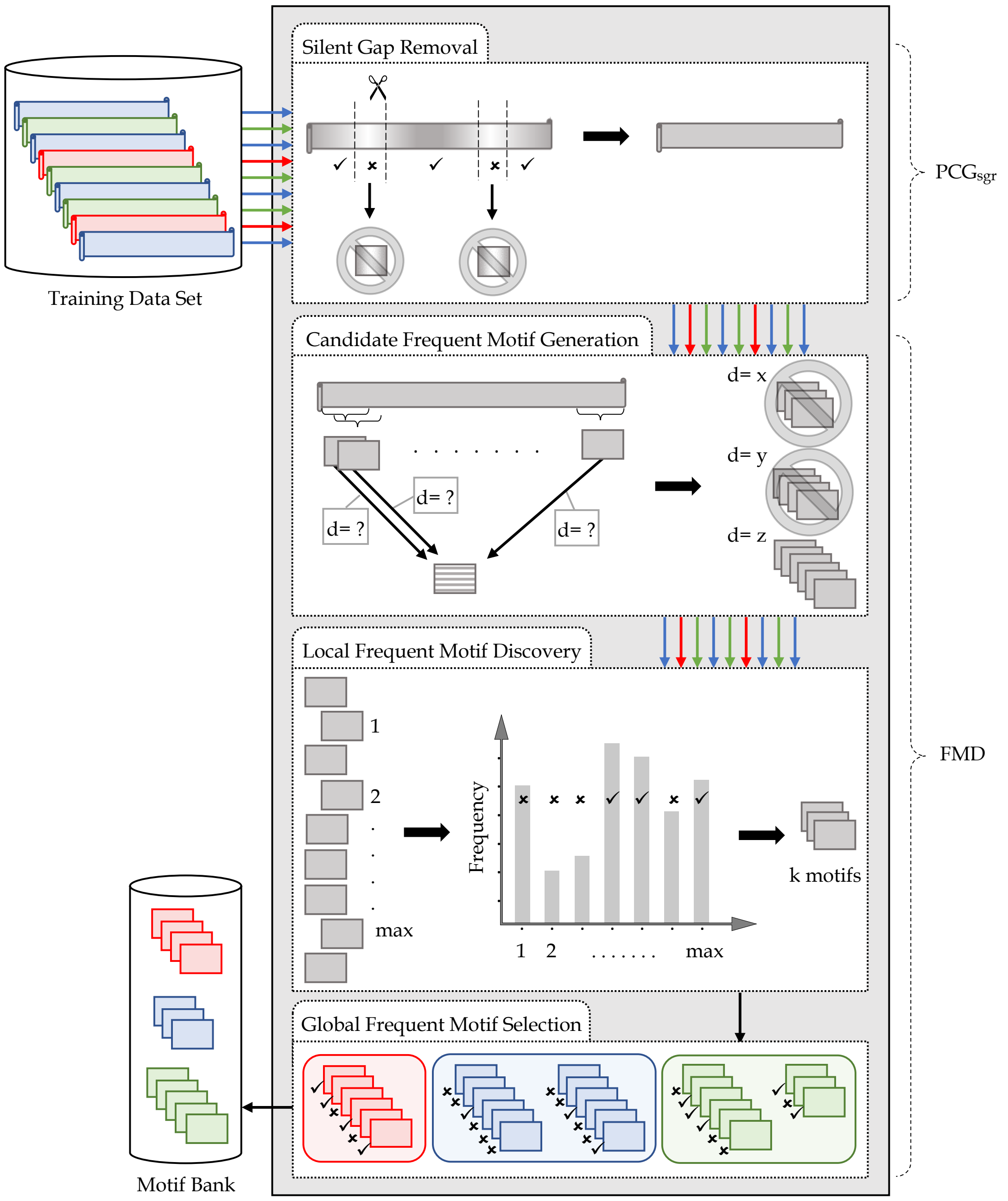


Figure 7.1: Schematic showing the basic operations of the SGR-FMD approach

ing to their class-label, and then each "local" motif is checked globally against all motifs in the different classes; if there is no similar motif in the time series belonging to other classes, it is considered "global" and marked with a "✓" (good discriminator of class), while the remainder are marked with an "**x**" (not good discriminators of class). The retained motifs, those with check-marks, then form the "motif bank" to be used to label previously unseen time series.

The pseudo code for the parent process is given in Algorithm 14. The inputs are: (i) a set T of point series and class pairs, (ii) a window size ω, (iii) the maximum size max for specifying the number of sub-sequences in Q', (iv) a threshold k for limiting the number of frequent motifs selected, (v) a threshold α for pruning infrequent sub-sequences, (vi) a threshold σ_m for defining the concept of frequency and (vii) a threshold λ for defining the concept of similarity. The output is a set D' of frequently occurring motifs that are considered to be good global discriminators of class.

Algorithm 14 SGR-FMD Approach

```
Require: $T$, $\omega$, $max$, $k$, $\alpha$, $\sigma_m$, $\lambda$
Ensure: $D'$
 1: $D \leftarrow \emptyset$, A set to hold frequent motifs that are good local discriminators of class
 2: for $\forall\langle P_i, c_i\rangle \in T$ do
 3:     $P'_i \leftarrow PCG_{sgr}(P_i)$
 4:     $Q_i \leftarrow$ A set of sub-sequences of length $\omega$ in $P'_i$
 5:     $Q'_i \leftarrow candidateFrequentMotifGen(Q_i, \alpha)$
 6:     $max = \frac{max}{|Q'_i|}$ , $k = \frac{k}{|Q'_i|}$
 7:     for $\forall Q'_{i_j} \in Q'_i$ do
 8:         $L \leftarrow localMotifDiscovery(Q'_{i_j}, max, k, \sigma_m, \lambda, c_i)$
 9:         $D \leftarrow D \cup L$
10:     end for
11: end for
12: if $D \neq \emptyset$ then
13:     $D' \leftarrow globalMotifSelection(D)$
14: end if
15: return( $D'$ )
```

The algorithm commences by initialising the set D with $\emptyset$ (empty set). It then processes each time series and class pair $\langle P_i, c_i\rangle$ in T in turn. The size of each time series P_i is first reduced (line 3) by removing "silent gaps" so as to give a time series P'_i. Next, the set Q_i is generated (line 4) comprised of the complete set of sub-sequences in P'_i of length ω. This is then further processed (line 5) to identify a set of θ sets of candidate frequent motifs Q'_i; in other words, sub-sequences that are infrequent are removed. The threshold max and k are then calculated in line 6. Up to max frequent candidate motifs are retained, through a call to the process *localMotifDiscovery*, and the set D is populated, the set of frequently occurring motifs that are good local discriminators. The set D, if not empty, is processed further in line 13 so that only motifs that are good global class discriminators are retained; these are held in a set D'. The motifs in the set D' are then returned (line 15) to be used as a “motif bank” in (say) a Nearest Neighbour Classification model with which to label previously unseen time series.

From Algorithm 14, it can be seen that there are four function calls. The first call (line 3) is for the Silent Gap Removal pre-processing, PCG$_{\text{sgr}}$, which is presented in the following section, Section 7.3. The remaining three function calls (lines 5, 8 and 13) are for the three sub-processes of the FMD algorithm, these are discussed in further detail in Section 7.4.

7.3 Silent Gap Removal

This section presents the PCG$_{\text{sgr}}$ silent gap removal technique. A number of pre-processing methods were presented in Chapter 2. It was noted that, in some domains, it is possible to prune (pre-process) time series data by removing parts that are previously known to be irrelevant to the application; the techniques used for this purpose were referred to as exclusive techniques. In the case of Amplitude-PCG data, a particular feature is that of “silent gaps” existing in heartbeat sequences. An exclusive technique could be used with respect to this data, given the domain knowledge, to exclude these gaps which are known not to carry much information and appear in almost every PCG signal.

A feature of PCG signals is that they comprise cycles, each representing a heartbeat, possibly some murmurs if diseased, and “silent gaps”. These silent gaps, because they appear in all time series held in T, can be assumed not to contribute to any form of class discrimination and therefore can be removed. The concept of silent gaps was originally identified in the context of processing voice recordings [28, 67, 127, 141, 161, 179]. The adopted mechanism for removing silent gaps is founded on the MathWorks mechanism[1] for removing silent gaps in such recordings. The mechanism of removing silent gaps from PCG signals, PCG$_{\text{sgr}}$, operates using a sliding window, of a pre-specified length, to identify a collection of all non-overlapping sub-sequences, Q, in a PCG signal.

For each sub-sequence, q_i in Q, two parameters are calculated: (i) the signal energy (e_i) and (ii) the spectral centroid (n_i). The first, e_i, is the mean squared value of the Amplitude values (p) in the i^{th} sub-sequence; calculated as shown in Equation 7.1.

$$e_i = \frac{\sum_{j=1}^{|q_i|} p^2_{(i\times|q_i|)-|q_i|+j}}{|q_i|} \tag{7.1}$$

The parameter n_i is the centre of “mass” of its spectrum (spectral centroid) obtained using the weighted Discrete Fourier Transform (DFT) coefficient of the Amplitude values (p) in the i^{th} sub-sequence and calculated as in Equation 7.2.

$$n_i = \frac{\sum_{j=1}^{|q_i|} (j+1)\times DFT_{(i\times|q_i|)-|q_i|+j}}{\sum_{j=1}^{|q_i|} DFT_{(i\times|q_i|)-|q_i|+j}} \tag{7.2}$$

[1]https://uk.mathworks.com/matlabcentral/fileexchange/28826-silence-removal-in-speech-signals

The pseudo code for the PCG$_{\text{sgr}}$ process presented above is given by Algorithm 15. The inputs are: (i) a point series P and (ii) a window length win measured in milliseconds. The output is a pruned point series P'. The process commences (line 1) by segmenting the input point series into a set of non-overlapping sub-sequences (frames) Q. The parameters e_i and n_i are then computed for each sub-sequence q_i in Q and stored in the sets E and N respectively (lines 5 to 10). The thresholds, t_e and t_n, required to identify a sequence as representing a silent gap are then calculated on lines 11 and 12; the threshold calculation process is given in Algorithm 16 described below. Then (lines 13 to 17) for each sub-sequence q_i in Q, if both of its parameters (e_i and n_i) are greater than or equal to the thresholds t_e and t_n, the sub-sequence is appended to P', the set of retained sub-sequences so far. The process continues until all the sub-sequences in Q have been processed and either retained or rejected. The pruned point series P' is then returned (line 18).

Algorithm 15 PCG$_{\text{sgr}}$

```
Require: $P$, $win$
Ensure: $P'$
 1: $Q \leftarrow$ A set of non-overlapping sub-sequences of length $win$ in $P$
 2: $P' \leftarrow \emptyset$, An empty set to hold pruned point series $P$
 3: $E \leftarrow \emptyset$, An empty set to hold signal energy values
 4: $N \leftarrow \emptyset$, An empty set to hold spectral centroid values
 5: for $\forall q_i \in Q$ do
 6:     $e_i \leftarrow$ Signal energy calculated using Equation 7.1
 7:     $E \leftarrow E \cup e_i$
 8:     $n_i \leftarrow$ Spectral centroid calculated using Equation 7.2
 9:     $N \leftarrow N \cup n_i$
10: end for
11: $t_e \leftarrow findThreshold(E)$
12: $t_n \leftarrow findThreshold(N)$
13: for $i = 1$ to $|E|$ do
14:     if $e_i \geq t_e$ and $n_i \geq t_n$ then
15:         $P' \leftarrow append(P', q_i)$
16:     end if
17: end for
18: return( $P'$ )
```

The pseudo code for the function *findThreshold*, called from lines 11 and 12 in the PCG$_{\text{sgr}}$ (Algorithm 15), is given in Algorithm 16. The input is a collection of values F; the output is a calculated threshold t. The process uses a smoothing window parameter μ_e which is recommended[2] to be set to 5. The first step (line 3) is to generate the histogram G; this is then (line 4) smoothed, using the parameter μ_e, giving a smoothed histogram G'. The local maxima in G' are then identified and stored in a set $A = \{a_1, a_2, \dots\}$ (line 5). The threshold t is then calculated according to the number of maxima in A (lines 6 to 11), which is then returned (line 12).

[2]https://uk.mathworks.com/matlabcentral/fileexchange/28826-silence-removal-in-speech-signals

Algorithm 16 function findThreshold

```
Require: $F$
Ensure: $t$
 1: $avg$, A variable to hold the average of a histogram values
 2: $\mu_e \leftarrow 5$, The “smoothing window” size
 3: $G \leftarrow$ The histogram of $F$
 4: $G' \leftarrow$ Smoothing filter applied to $G$ using parameter $\mu_e$
 5: $A \leftarrow$ Local maxima in $G'$
 6: if $|A| > 1$ then
 7:     $t \leftarrow \frac{\mu_e \times (a_1 + a_2)}{\mu_e + 1}$
 8: else
 9:     $avg \leftarrow$ The average value of $G$
10:     $t \leftarrow \frac{avg}{2}$
11: end if
12: return( $t$ )
```

7.4 Frequent Motif Detection

In Section 7.2, a high-level overview of the approach was presented. In the previous section, Section 7.3, the operation of Silent Gap Removal, PCG$_{\text{sgr}}$, was presented. Once PCG$_{\text{sgr}}$ has been applied to all the point series in the collection to be considered, the next step is the discovery (mining) of frequent motifs, the FMD presented in this section.

As discussed earlier in Chapter 2, a motif is a time series sub-sequence that is representative of a class and can therefore be used for the purpose of time series classification. Determining whether a motif is representative of a time series or not is explored with respect to this thesis using two strategies: (i) similarity- and (ii) frequency-based. With respect to the motif discovery algorithm used in the previous two chapters, Chapters 5 and 6, the similarity-based strategy was used. An alternative strategy of extracting motifs is based on frequency, which is the technique that was adopted with respect to the proposed SGR-FMD approach presented in this chapter. This technique, the Frequent Motif Detection (FMD) technique, is therefore presented in this section.

Given a set of sub-sequences Q, a frequent motif is a sub-sequence $q \in Q$ that occurs in Q more often than some threshold σ_m. In other words, for q_j to be considered to be a frequent motif, the set Q must include at least σ_m sub-sequences that are in some sense all similar to q_j, as defined according to a similarity threshold λ. The set of frequent motifs generated from Q is then given by $M = \{\langle m_1, f_1\rangle, \langle m_2, f_2\rangle, \dots\}$, where f_i is the frequency count ($f_i \geq \sigma_m$).

Given a set of frequent motifs M, the k most frequently occurring motifs associated with a class-label c_i, arguably, are the most representative of the underlying class associated with the original time series P. The top k motifs drawn from a set of frequent motifs M is given by the set $L = \{\langle m_1, c_i\rangle, \langle m_2, c_i\rangle, \dots, \langle m_k, c_i\rangle\}$. Given a collection of time series T, the complete set of identified frequent motifs that exist in the time series collection is then given by $D = \bigcup_{i=1}^{z} L_i$, where z is the number of records in T. Each motif in D is included because it was considered to be a good *local* representative of a class. This does not necessarily mean that it is also a good *global* representative. The set D therefore needs to be pruned to give D', a motif set that contains only motifs that are good global representatives. The complement of D' in D is then the set D'', the set of motifs that are not good global discriminators of class. A good representative motif is defined as one where either there are no similar motifs in D or if there are similar motifs in D, they are all linked to the same class; D' and D'' can be more formally defined as follows:

- $D', D'' \subset D \ : \ D' \cup D'' = D$ and $D' \cap D'' = \emptyset$.
- $\forall \ d_i \in D' \nexists \ d_j \in D \ : \ m_i \simeq m_j \ \wedge \ c_i \neq c_j$.
- $\forall \ d_i \in D'' \exists \ d_j \in D \ : \ m_i \simeq m_j \ \wedge \ c_i \neq c_j$.

As mentioned above, the proposed motif discovery algorithm, FMD, comprises three sub-processes as follows:

1. Candidate Frequent Motif Generation.
2. Frequent Motif Discovery for Local Class Discrimination.
3. Frequent Motif Selection for Global Class Discrimination.

Each sub-process is detailed in the following three sub-sections, Sub-sections 7.4.1 to 7.4.3.

7.4.1 Candidate Frequent Motif Generation

Given a pruned point series P', with silent gaps removed, this can be pruned further by removing sub-sequences that clearly cannot be considered to be frequent so as to retain a set of candidate frequent motifs Q', where Q' is a subset of Q, the complete set of sub-sequence in P' of length ω. Note that this does not mean that all sub-sequences in Q' will be frequent, it simply means they are likely to be frequent. To do this, a novel algorithm was proposed whereby a hypothetical motif $zm = \{zm_j \ : \ zm_j = 0, \ 1 \leq j \leq \omega\}$ was used. This hypothetical motif is referred to as the "zero-motif" for obvious reasons. The similarity between each sub-sequence $q_i = \{q_{i_1}, q_{i_2}, \ldots, q_{i_\omega}\}$, $q_i \in Q$, and zm is calculated using a similarity function; Euclidean Distance was used with respect to the evaluation presented later in this chapter, calculated as shown in Equation 7.3:

$$d(q_i, zm) = \sqrt{\sum_{j=1}^{\omega} (q_{i_j} - zm_j)^2} \tag{7.3}$$

However, given that the zero-motif is a vector of zeros, the similarity function can be simplified as presented in Equation 7.4, which coincides with the Root Sum Square (RSS) method.

$$d(q_i, zm) = d_{z_i} = \sqrt{\sum_{j=1}^{\omega} q_{i_j}^2} \tag{7.4}$$

RSS assumes that most of the point values q_i fall in the middle of the Q value range; in other words, the point values could be easily clustered. The obtained distance values $d(q_i, zm)$ were therefore used to cluster the set of sub-sequences Q into an ordered set of clusters, $CL = \{CL_1, CL_2, \dots\}$, one cluster per unit distance value, in descending order according to size. The sub-sequences in Q contained in the largest cluster, CL_1, were retained; however, if the difference between the size of CL_1 and CL_2 was proportionally small, then the sub-sequences represented by the second cluster were also retained. This is dynamically computed; whether one or two clusters are retained was defined by a parameter θ, which can take the value 1 or 2 and was calculated using Equation 7.5 as follows:

$$\theta = \begin{cases} 1 & \text{if } \frac{|CL_1| - |CL_2|}{|Q|} \times 100 > \alpha \\ 2 & \text{otherwise} \end{cases} \tag{7.5}$$

Where, α is a percentage of the total number of sub-sequences in Q. This means that if CL_1 contains $x\%$ of the sub-sequences in Q, CL_2 must contain at least $x - \alpha\%$ of the sub-sequences in Q to be retained; otherwise, only CL_1 is retained. This closeness in size between the largest two clusters is important because it indicates the significance of the clusters. The value $\alpha = 5\%$ was used with respect to the evaluation presented in Section 7.5 because $alpha = 0.05$ is frequently used in statistical analyses as the cut-off for significance [86].

The pseudo code for the Candidate Frequent Motif Generation sub-process is given in Algorithm 17. The inputs are: (i) a set of sub-sequences Q and (ii) a threshold α. The output is a set Q' of θ sets of candidate frequent motifs; the number of sets is dynamically computed according to the clustering results as illustrated above, which may consist of one or two sets.

The algorithm commences (line 1) by defining the zero-motif zm, this is included for clarity purposes, in practice zm does not need to be specifically defined. Next, an empty set D_z corresponding to the set Q is initialised (line 2) to hold the similarity values once calculated. The set Q is then processed (lines 3 to 6) so as to populate D_z; note that there is a one-to-one correspondence between Q and D_z. The set Q is then clustered (line 7) according to D_z, each cluster represents a unit distance. The largest cluster is retained (line 8) and the second largest might also be retained depending on the θ parameter value (lines 9 to 12). The retained set of clusters Q' is then returned (line 13).

7.4.2 Frequent Motif Discovery for Local Class Discrimination

The next sub-process, *localMotifDiscovery*, takes a set of candidate frequent motifs Q', and identifies the most frequent motifs that, by definition, are deemed to be good local discriminators. However, even after silent gap removal and the removal of infrequent

Algorithm 17 Candidate Frequent Motif Generation

```
Require: $Q$, $\alpha$
Ensure: $Q'$
 1: $zm \leftarrow$ The zero-motif
 2: $D_z \leftarrow \emptyset$, A set to hold distance values
 3: for $\forall q_i \in Q$ do
 4:     $d_{z_i} \leftarrow$ The euclidean similarity between $q_i$ and $zm$ calculated using Equation 7.4
 5:     $D_z \leftarrow D_z \cup d_z$
 6: end for
 7: $CL \leftarrow$ An ordered set of clusters obtained by clustering all $q_i \in Q$ according to the
    corresponding value $d_{z_i} \in D_z$, $CL = \{CL_1, CL_2, \ldots\}$
 8: $Q' \leftarrow CL_1$
 9: $\theta \leftarrow$ A parameter calculated using Equation 7.5
10: if $\theta = 2$ then
11:     $Q' \leftarrow Q \cup CL_2$
12: end if
13: return( $Q'$ )
```

sub-sequences, as described above, the number of remaining sub-sequences in Q' is still likely to be large. We therefore propose limiting the number of candidate frequent motifs considered using a user-defined threshold max. Thus, the idea is to randomly select max candidates from the set Q'.

The pseudo code for the Local Frequent Motif Discovery algorithm is presented in Algorithm 18. The inputs are: (i) a set of candidate frequent motifs Q', (ii) the thresholds max, k, σ_m and λ and (iii) a class-label c associated with the given time series. The output is a set L of identified frequent motifs.

The algorithm commences (lines 1 and 2) by defining the sets M and L to hold the selected motifs and, line 3, a counter to ensure that max candidate motifs are selected. The process then enters a loop (lines 4 to 14) in which up to max time series are selected. On each iteration, a candidate time series sub-sequence q_i is randomly selected from Q' (line 5). In the case where the set Q' is empty, or all its sub-sequences have been tested, this loop will be terminated (lines 6 to 8) even though max candidate motifs have not been added to M; otherwise, the frequency count f_i for the sub-sequence q_i is calculated (using an Euclidean Distance function and the λ parameter for determining the similarity between sub-sequences while maintaining the early abandonment concept). In each case, if the count is greater than or equal to σ_m% of $|Q'|$, the sub-sequence and its frequency value are stored in the set $M = \{\langle m_1, f_1\rangle, \langle m_2, f_2\rangle, \ldots\}$. Note that on completion of the loop, if a high σ_m threshold value has been used, the set M may be empty, this is tested for in line 15. If the set M is not empty, it is processed by testing its size, if it is greater than k, the content of the set M is arranged in descending order according to the frequency values (line 17). If the size of M is less than k, the latter is adjusted to $|M|$ (line 19). Next, the set L of the k most frequently occurring motifs, paired with their class-label, is generated (lines 21 to 23). The set L is then returned (line 25).

Algorithm 18 Local Frequent Motif Discovery

```
Require: $Q'$, $max$, $k$, $\sigma_m$, $\lambda$, $c$
Ensure: $L$
 1: $M \leftarrow \emptyset$, An empty set to hold motifs
 2: $L \leftarrow \emptyset$, An empty set to hold top $k$ frequent motifs
 3: $count \leftarrow 0$, A counter variable
 4: while $count < max$ do
 5:    $q_i \leftarrow$ A randomly selected sub-sequence from $Q'$
 6:    if $q_i =$ Null then
 7:       exit this loop
 8:    end if
 9:    $f_i \leftarrow$ The number of sub-sequences in $Q'$ that are similar to $q_i$ according to the
       threshold $\lambda$
10:    if $f_i \geq \frac{\sigma_m \times |Q'|}{100}$ then
11:       $M \leftarrow M \cup \langle q_i, f_i \rangle$
12:       $count \leftarrow count + 1$
13:    end if
14: end while
15: if $M \neq \emptyset$ then
16:    if $|M| > k$ then
17:       Reorder the set $M$ according to frequency values in descending order
18:    else
19:       $k \leftarrow |M|$
20:    end if
21:    for $i = 1$ to $k$ do
22:       $L \leftarrow L \cup \langle q_i, c \rangle \quad (\langle q_i, \_ \rangle \in M)$
23:    end for
24: end if
25: return( $L$ )
```

The Silent Gap Removal pre-process, and both Candidate Frequent Motif Generation and Local Frequent Motif Discovery sub-processes, are applied to each point series in the input set $T = \{P_1, P_2, \dots\}$; thus, a sequence of sets L are generated, $\{L_1, L_2, \dots\}$. These are collated into a set D as shown previously in line 9 in the parent process, which was given in Algorithm 14.

7.4.3 Frequent Motif Selection for Global Class Discrimination

As noted earlier, although the individual motifs in D were deemed to be good local class discriminators, this did not automatically mean that they were also good global class discriminators. The issue here was that D may contain pairs, $\langle m_i, c_i \rangle$ and $\langle m_j, c_j \rangle$, where $m_i \simeq m_j$ and $c_i \neq c_j$. In other words, D may contain motif-class pairs that contradict each other. Thus, the final step was to derive a set $D' \subset D$ that comprised only good global class discriminators within the context of the time series collection T. The implementation of this step is presented in this sub-section.

The pseudo code for the Global Frequent Motif Selection process is presented in Algorithm 19. It takes as input: (i) the generated set of frequent motifs D and (ii) a threshold λ for similarity calculation. The output is the set D' which contains the motifs that are good global discriminators.

Algorithm 19 Global Frequent Motif Selection

```
Require: $D$, $\lambda$
Ensure: $D'$
 1: $D' \leftarrow \emptyset$, A set to hold frequent motifs that are good global discriminators of class
 2: *isGoodGlobalDiscriminator*, A Boolean variable used as a flag to detect not global discriminators
 3: for $\forall \langle m_i, c_i \rangle \in D$ do
 4:     *isGoodGlobalDiscriminator* $\leftarrow$ true
 5:     for $\forall \langle m_j, c_j \rangle \in D$ and $i \neq j$ do
 6:         if $c_i \neq c_j$ and $m_i \simeq m_j$ then
 7:             *isGoodGlobalDiscriminator* $\leftarrow$ false
 8:             exit this loop
 9:         end if
10:     end for
11:     if *isGoodGlobalDiscriminator* then
12:         $D' \leftarrow D' \cup \langle m_i, c_i \rangle$
13:     end if
14: end for
15: return( $D'$ )
```

The Algorithm commences by defining a variable *isGoodGlobalDiscriminator*, a Boolean variable used as a flag to detect global discriminators; it is assumed, in the beginning of the search, that each motif is a good global discriminator (line 4) until the alternative is established (line 7). In the Global Frequent Motif Selection sub-process, each motif-class pair in D (line 3) is compared with all other motif-class pairs in D (line 5); if no similar motif associated with a different class-label is found (lines 6, 7 and 11), the motif-class pair is then added to the set D' (line 12). Similarity is measured in the same way as before using the Euclidean Distance function and the λ parameter. Checking all motif-class pairs in D, the data set D' is returned (line 15) which can then be used as a "motif bank" for the Nearest Neighbour Classification (NNC) for labelling previously unseen time series.

7.5 Evaluation

In the previous two sections, Sections 7.3 and 7.4, the operation of the Silent Gap Removal technique, PCG_{sgr}, and Frequent Motif Detection algorithm, FMD, were presented. These two algorithms form the SGR-FMD approach which is evaluated in detail in this section. The evaluation results obtained with respect to a set of experiments undertaken, using the collected evaluation data set (see Chapter 3), to analyse the operation of the proposed approach were considered. The objectives of the evaluation were:

Objective 1, Operational Analysis: To investigate the operation of the proposed approach.

Objective 2, Comparison of Pruning Techniques: To compare the effect of the pruning techniques used in SGR-FMD approach on the data size and on the classification performance.

Objective 3, Comparison with the $\mathbf{PCG_{seg}}$ Classification Approach: To compare the operation of SGR-FMD approach with the PCG_{seg} Classification approach described in the previous chapter in terms of computational efficiency (runtime) and effectiveness (accuracy).

Objective 4, Most Appropriate Parameter Settings: To establish the most appropriate parameter settings for:

- The Silent Gap Removal pre-processing technique (PCG_{sgr}), which used a window length win expressed in milliseconds.
- The Frequent Motif Detection algorithm (FMD), which used the parameters: (i) ω, the size of the candidate motifs, (ii) λ, the similarity threshold, (iii) σ_m, the frequency threshold and (iv) k and max, the selection thresholds.
- The used classification model (NNC) and the most appropriate method for conflict resolution, where a number of motifs is considered.

Each of the above objectives are considered in turn in the following four sub-sections, Sub-sections 7.5.1 to 7.5.4. The evaluation metrics recorded were accuracy (acc), precision (prec), recall (rec), F-score (f-s) and runtime. Five-fold cross-validation was adopted and the data was statically stratified with respect to all the reported experiments. The evaluation was conducted using the Java programming language. The experiments were run on an iMac Pro (2017) computer with 8-Cores, 3.2GHz Intel Xeon W CPU and 19MB RAM.

7.5.1 Operational Analysis

In this sub-section, the evaluation of the operation of the SGR-FMD approach is presented. Examples of Amplitude-PCG motifs, for each of the four classes, obtained during the evaluation of the SGR-FMD approach are given in Figure 7.2. Using NNC and five-fold cross-validation, the best and worst accuracies recorded were 71.3% and 56.1%. The precisions, recalls and F-scores were 0.458, 0.442 and 0.440 for the best accuracy, and 0.112, 0.166 and 0.124 for the worst. In addition, the recorded standard deviation for the classification model was good, with an average of around 0.05. The average runtime to extract the k most frequent motifs from one record was 2.5 seconds. The total runtime for the whole data set was about two and a half minutes, on average, for each experiment; a set of nine experiments was conducted to determine the best values for the FMD algorithm variables. More details are given in Sub-section 7.5.4.

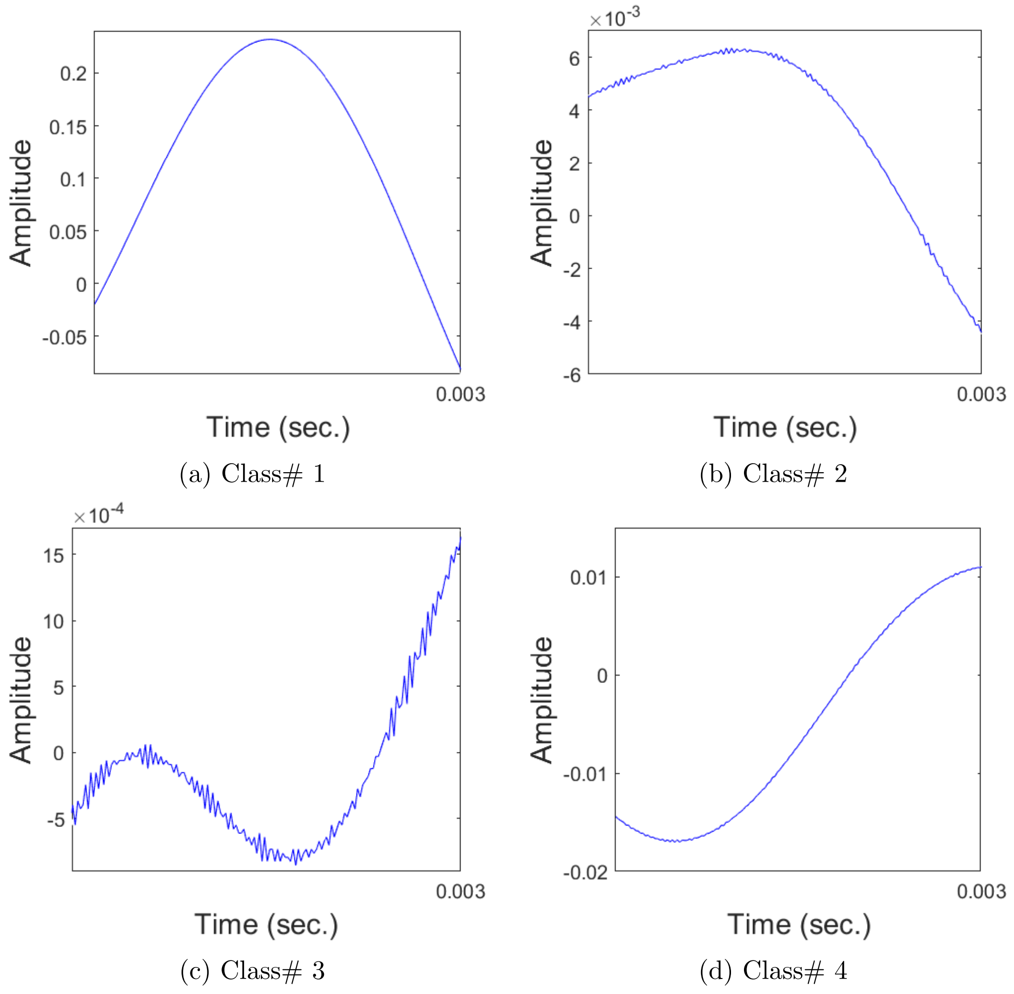


Figure 7.2: Example motifs for each class in the Amplitude-PCG evaluation data generated using the SGR-FMD approach

7.5.2 Comparison of Pruning Techniques

In the proposed SGR-FMD approach, two pruning techniques were used: (i) Silent Gap Removal using the PCG_{sgr} algorithm, and (ii) excluding infrequent sub-sequences during Candidate Frequent Motif Generation. Applying the former technique, the number of points in each input time series P_i was reduced by just under half (47%). The Candidate Frequent Motif Generation process then took as input a pruned point series P'_i (the output from the PCG_{sgr}) and a window size ω defining the motif size (in terms of a number of points). Experiments were conducted using $\omega = \{100, 200, 300\}$ to identify the most appropriate parameter settings (more details are given in Sub-section 7.5.4). It was found that the motif size affects the percentage of retained points. The results indicated that when $\omega = 200$, the point series size was reduced by approximately a further 45% (71% of the original size); when $\omega = 100$ and $\omega = 300$, the point series size was reduced by approximately a further 32% (64% of the original size). It would appear, from a pruning only perspective, that $\omega = 200$ was the most effective.

Given that the raw Amplitude-PCG time series comprised approximately 50 million data points, this was reduced to 26.5 million data points once PCG_{sgr} had been applied, and down to between 14.5 and 18 million data points (depending on the value for ω used) once infrequent sub-sequences had been removed; a substantial difference from 50 million data points. This reduction in size was anticipated to be reflected in the execution time; althoughthe impact of each of the two techniques on the classification accuracywas unknown.

Further experiments were conducted to determine the effect on runtime and accuracy when either the Silent Gap Removal or the Candidate Frequent Motif Generation phase was omitted, and when both were omitted. In the first case, line 3 in Algorithm 14 was removed and line 4 was replaced with:

$$Q_i \leftarrow \text{A set of sub-sequences of length } \omega \text{ in } P_i$$

In the second case, line 4 in Algorithm 14 was replaced with:

$$Q'_i \leftarrow \text{A set of sub-sequences of length } \omega \text{ in } P'_i$$

and lines 5, 6, 7 and 10 were removed. In the third case, lines 3, 4 and 5 in Algorithm 14 were replaced with:

$$Q'_i \leftarrow \text{A set of sub-sequences of length } \omega \text{ in } P_i$$

and lines 6, 7 and 10 were removed.

The results obtained by these three sets of experiments are given in Appendix B in Tables B.1 to B.3 for classification performance, and Table B.4 for runtime. The results using the best performing parameters are presented in Table 7.2. The accuracy results are average results obtained using five-fold cross-validation and the runtime is given in seconds (and decimals of seconds). Considering PCG_{sgr} first, the first two rows in the table, it can be seen that PCG_{sgr} had a slight adverse effect on accuracy. However, it improved runtime although this is not entirely apparent from the reported runtime in the table because different ω and k values produced the best results (recall that low ω and k values result in efficiency gains because they entail less calculation). Candidate Frequent Motif Generation, on the other hand, had a positive effect on accuracy and resulted in significant speed up (although again it should be noted that the results reported in Table 7.2 were obtained using different ω values). When the two were run together, as in the case of later experiments reported in Sub-section 7.5.4.3, accuracy was slightly reduced, because of the negative effect of PCG_{sgr}, but runtime was enhanced considerably.

7.5.3 Comparison with the PCG_{seg} Classification Approach

To evaluate the performance of the SGR-FMD approach presented in this chapter, its operation was compared with the PCG_{seg} Classification approach presented in Chapter 6,

Table 7.2: The best classification accuracy for finding frequent motifs with different pruning techniques

Pruning Techniques		Attributes				Results	
PCG_{sgr}	CFMG	ω	k	CRM	k_{nnc}	Acc	Runtime
x	x	300	10	HV	3	0.703	08.27
✓	x	100	30	HV	1	0.693	15.39
x	✓	300	10	SD	3	0.716	03.74
✓	✓	100	10	HV	1	0.713	00.49

Key: CFMG = candidate frequent motif generation, the first step of FMD; and CRM = conflict resolution method.

the best approach presented in this thesis so far, in terms of computational efficiency and effectiveness. To determine the runtime complexity, nine sets of experiments were conducted using $\omega = \{100, 200, 300\}$ and $k = \{10, 20, 30\}$. The adopted values for both λ and σ_m were 0.025 because the experiments reported in Sub-section 7.5.4 had indicated that these were the most appropriate "general" values to use; $max = 2 \times k$ was used.

The recorded runtime results are presented in terms of milliseconds in Table 7.3, these are average runtimes for one PCG time series obtained by running each experiment five times (FCV). In the table, the runtimes for the SGR-FMD approach are presented for the Silent Gap Removal pre-process (PCG_{sgr}) and the Frequent Motif Detection process (FMD) which are divided into three sub-process: (i) Candidate Frequent Motif Generation, (ii) Local Frequent Motif Discovery and (iii) Global Frequent Motif Selection. The last column in the table presents the total runtime to process a single PCG time series (the sum of the values in the previous four columns).

Table 7.3: Runtime results for the SGR-FMD approach (in milliseconds)

ω	k	PCG_{sgr}	FMD			Total
			CFMG	LFMD	GFMS	
100	10	55.76	116.61	0316.27	0.17	0488.81
	20			0634.92	1.69	0808.98
	30			0952.54	5.42	1130.33
200	10		094.75	1021.86	0.68	1173.05
	20			2040.68	2.88	2194.07
	30			3062.20	8.14	3220.85
300	10		083.56	2190.85	0.85	2331.02
	20			4376.10	3.39	4518.81
	30			6560.34	9.32	6708.98

Key: CFMG = candidate frequent motif generation; LFMD = local frequent motif discovery; and GFMS = global frequent motif selection.

From the table, it is evident that the larger the ω value (the window size), the less time that was required to generate a set of candidate frequent motifs because when using a large ω value, there will be fewer sub-sequences to consider. However, the runtime required for the Local Frequent Motif Discovery sub-process increases with ω, because larger sub-sequences require more processing. Note that the runtime required for the final Global Frequent Motif Selection sub-process is negligible.

To compare with the PCG_{seg} Classification approach in terms of efficiency, the runtime required for both approaches are presented in Table 7.4, these are the total runtimes, including pre-processing, to classify one record in mm:ss.SS format. From the table, it can be clearly seen that the runtime for the SGR-FMD approach was much better than the time required with respect to the PCG_{seg} Classification approach. It took, on average, two and a half seconds to process a record using the SGR-FMD approach, this compared very favourably with the PCG_{seg} Classification approach which required more than 11.5 minutes (about 698 seconds) to process a record. The proposed SGR-FMD approach therefore reduced the model generation time by a factor of more than 278.

Table 7.4: Recorded runtime results (mm:ss.SS format) for the SGR-FMD and PCG_{seg} Classification approaches

ω	k	SGR-FMD Approach	ω	r	PCG_{seg} Approach
100	10	00:00.49	25	2	10:02.03
	20	00:00.81		4	06:04.15
	30	00:01.13		6	13:51.10
200	10	00:01.17	50	2	09:01.25
	20	00:02.19		4	10:52.41
	30	00:03.22		6	07:00.39
300	10	00:02.33	75	2	18:03.06
	20	00:04.52		4	10:45.44
	30	00:06.71		6	19:08.09
Average		00:02.51	11:38.21		

To compare the SGR-FMD approach with the PCG_{seg} Classification approach in terms of effectiveness, the classification measurements for both approaches are presented in Table 7.5. For the proposed approach, only the experiments that used $k_{nnc} = 1$ with NNC coupled with a Highest Votes (HV) method to resolve class-label conflicts was used (where a number of motifs is considered, see discussion in Sub-section 7.5.4.3 below) are reported here because this combination produced the best classification measurements. The best recorded accuracy for the SGR-FMD approach was 71.3%, a reduction of 0.7% compared with the PCG_{seg} Classification approach which produced a best accuracy of 72.0%. The precision, recall and F-score of the SGR-FMD approach were 0.458, 0.442 and 0.440, this compared favourably with the PCG_{seg} Classification approach where 0.181, 0.281 and 0.211 were reported. Given that, using the best accuracy results, the SGR-FMD approach required only 0.49 seconds to mine one record, whereas the PCG_{seg} Classification approach required 7 minutes; and knowing that the application of this research is point-of-care disease diagnoses, the reduction in accuracy (0.7%) might be considered acceptable in order to gain a classification speed up by a factor of 858. Note that the classification is intended to be a guide and not a definitive result.

Table 7.5: Recorded accuracy, precision, recall and F-score for the SGR-FMD and PCG$_{\text{seg}}$ Classification approaches

SGR-FMD Approach						PCG$_{\text{seg}}$ Approach					
ω	k	Acc	Prec	Rec	F-s	ω	r	Acc	Prec	Rec	F-s
100	10	**0.713**	0.458	0.442	0.440	25	2	0.635	0.112	0.316	0.164
	20	0.686	0.150	0.226	0.172		4	0.583	0.041	0.250	0.065
	30	0.711	0.105	0.250	0.147		6	0.711	0.105	0.250	0.147
200	10	0.651	0.192	0.231	0.206	50	2	0.711	0.105	0.250	0.147
	20	0.645	0.197	0.253	0.220		4	0.661	0.142	0.203	0.165
	30	0.670	0.277	0.263	0.265		6	**0.720**	0.181	0.281	0.211
300	10	0.656	0.162	0.275	0.193	75	2	0.711	0.105	0.250	0.147
	20	0.636	0.158	0.236	0.185		4	0.670	0.151	0.246	0.186
	30	0.710	0.296	0.321	0.300		6	0.668	0.197	0.266	0.221

7.5.4 Most Appropriate Parameter Settings

As shown in the previous section, Section 7.4, the proposed approach used a number of parameters which needed to be set by the user. A set of values for each parameter was experimented with to determine the best value. These experiments are discussed in further detail, with respect to PCG$_{\text{sgr}}$, FMD and the classification model, in the following three sub-sections, Sub-sections 7.5.4.1 to 7.5.4.3 respectively.

7.5.4.1 Parameter Settings for the PCG$_{\text{sgr}}$ Technique

The silent gap removal technique used was originally designed for use with human-voice recordings; in this context, a window length of $win = 50$ milliseconds was suggested[3]. Experiments were conducted to determine the most appropriate value for win with respect to PCG signals. To this end, a range of window lengths from $win = 10$ milliseconds to $win = 90$ milliseconds, increasing in steps of 10 milliseconds, was experimented with. Examples of the results obtained are given in Figure 7.3 using a fragment of one of the Amplitude-PCG time series used for the evaluation. In the figure, sub-sequences that were retained are indicated in blue. The reason why some sub-sequences which clearly do not represent silent gaps have been discarded is because of the way the "signal energy" and "spectral centroid" parameters interact when large window lengths are used. From the figure, it can be seen that a window length of 10 milliseconds (last row) produced the best result. This was confirmed by visual inspection of further Amplitude-PCG example time series. This was the value therefore used with respect to the remainder of the experiments presented in this section.

[3]https://uk.mathworks.com/matlabcentral/fileexchange/28826-silence-removal-in-speech-signals

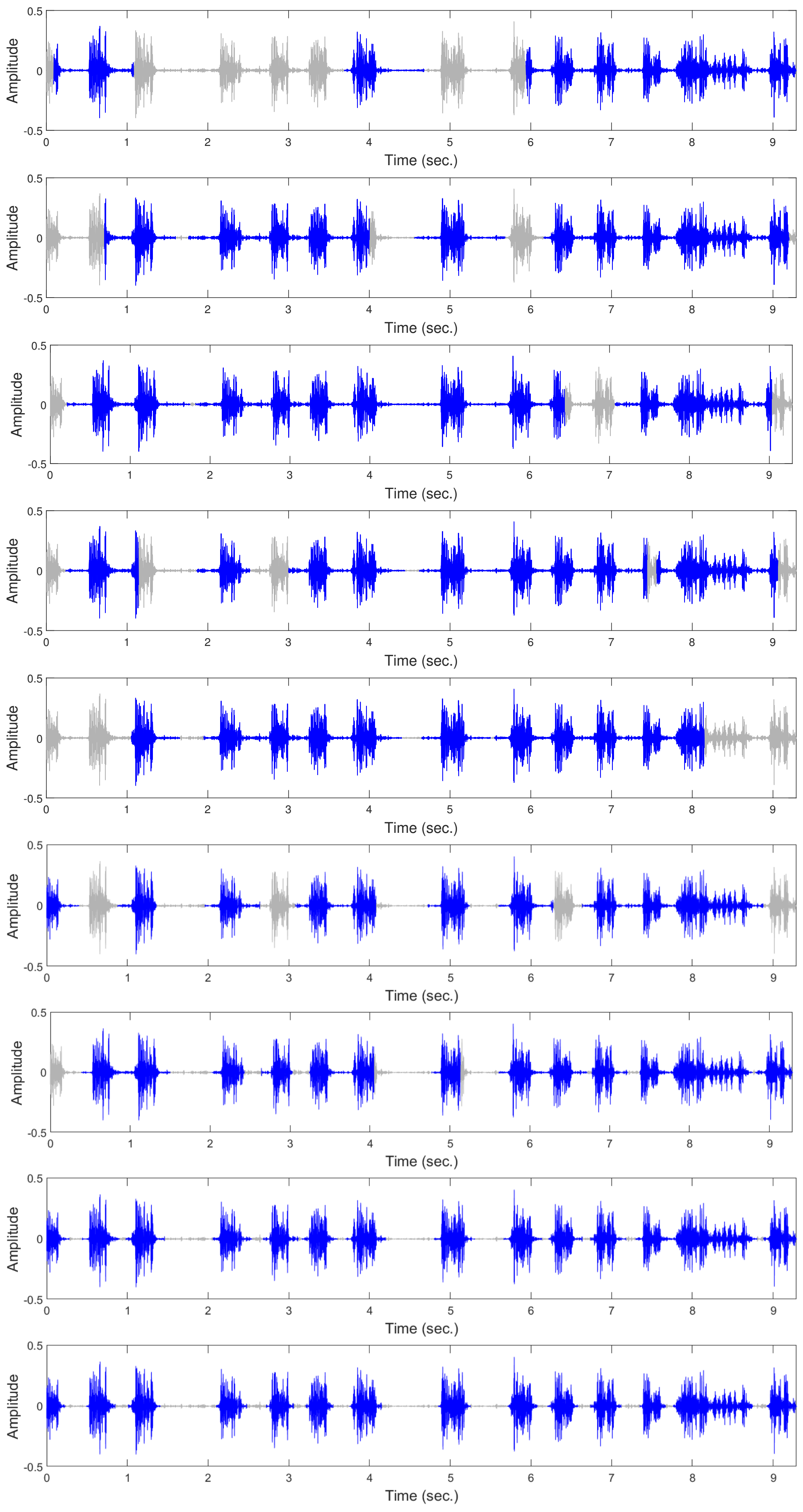


Figure 7.3: Sequence of Amplitude-PCG time series plots illustrating the operation of the PCG$_{sgr}$ technique using window lengths ranging from 90 to 10 milliseconds (top to bottom), greyed-out regions indicate sub-sequences that are identified as silent gaps and therefore pruned

7.5.4.2 Parameter Settings for the FMD Algorithm

Recall that the proposed Frequent Motif Detection algorithm required five parameters: (i) ω to define the desired frequent motif length expressed in terms of a number of points, (ii) max to define the maximum number of candidate frequent motifs to be considered, (iii) k to define the maximum number of frequent motifs to be selected ($k < max$), (iv) λ to determine the similarity threshold to be used, expressed in terms of the maximum distance between two motifs, (v) σ_m to determine the frequency threshold to be used, expressed in terms of the minimum percentage of possible motifs.

The selected values for these parameters all affect the number of frequent motifs identified and consequently the quality of any further utilisation of the motifs. Clearly, the higher the σ_m value, the fewer the number of motifs that would be identified because the criteria for frequency would become stricter as σ_m increased. Inversely, the higher the λ value, the greater the number of motifs that would be identified because the criteria for similarity would become less strict as λ increased; these parameters will be clarified more later on in this section. It was anticipated that as ω increased, the number of frequent motifs would decrease as there would be fewer sub-sequences to choose candidate frequent motifs from. The values for max and k would also affect the number of identified candidate frequent motifs and, it was conjectured, would thus also influence the number of selected frequent motifs as it will be mentioned later.

To identify the most appropriate parameter settings, a range of values for the parameters were considered although there was no guarantee that these selected values would yield a best result, they were intuitively selected for this purpose. The values for ω and k were $\{100, 200, 300\}$ and $\{10, 20, 30\}$ respectively. The value for max was set to $2 \times k$ although any value greater than k could have been used. The results obtained are given in Table 7.6. In the table, the results are presented for $k_{nnc} = 1$ and $k_{nnc} = 3$, and using three Conflict Resolution Methods (CRMs): (i) Shortest Distance (SD), (ii) Shortest Total Distance (STD) and Highest Votes (HV). Further consideration will be given to the value for k_{nnc} and the CRMs in the next sub-section, Sub-section 7.5.4.3. The best accuracy was obtained using $\omega = 100$ and $k = 10$ (71.3%), the grey cell in the table. The lowest accuracy score was obtained using $\omega = 100$ and $k = 30$. There does not seem to be any clear correlation between these two parameters and the obtained accuracy values.

Given the $\omega = 100$ had produced the best results, it was hypothesised that with even smaller window sizes, accuracy might increase further. Therefore, an extra experiment was conducted using $\omega = 50$ combined with the best performing parameter settings and conflict resolution method ($k = 10$ with $K_{nnc} = 1$ coupled with the HV conflict resolution method). However, it was found that accuracy dropped to 68.7%. It was also hypothesised, given that best results were obtained when $k = 10$, that accuracy might be improved if a lower value for k was considered. An extra experiment with $k = 5$ was therefore undertaken, with all other parameters as before, but it was found that this significantly reduced the accuracy to 51.4%.

Table 7.6: Classification performance measures using the SGR-FMD approach and different parameter settings

CRM	ω	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	100	10	0.693	0.310	0.356	0.329	**0.677**	0.234	0.266	0.246
		20	**0.686**	**0.192**	**0.248**	**0.213**	0.669	0.128	0.206	0.155
		30	**0.711**	**0.208**	**0.271**	**0.215**	**0.711**	**0.209**	**0.271**	**0.214**
	200	10	0.643	**0.278**	**0.319**	**0.297**	0.601	0.171	0.269	0.207
		20	0.636	**0.355**	0.231	**0.274**	0.637	**0.360**	0.244	**0.285**
		30	0.643	0.274	0.234	0.245	0.610	0.209	0.188	0.190
	300	10	0.654	**0.211**	**0.275**	**0.225**	**0.662**	**0.211**	**0.275**	**0.229**
		20	**0.636**	0.150	0.227	0.179	**0.650**	0.181	0.273	0.215
		30	0.657	**0.339**	**0.347**	**0.329**	0.644	0.251	0.293	0.261
STD	100	10	0.660	0.347	0.339	0.338	0.617	0.275	0.234	0.247
		20	0.602	0.190	0.218	0.202	0.626	**0.278**	**0.276**	**0.275**
		30	0.561	0.112	0.166	0.124	0.576	0.105	0.109	0.105
	200	10	0.635	0.213	0.289	0.244	**0.626**	**0.228**	**0.294**	**0.245**
		20	0.642	0.262	**0.301**	0.268	0.618	0.242	0.223	0.225
		30	0.575	0.165	0.124	0.134	0.593	0.244	0.214	0.223
	300	10	0.637	0.158	0.250	0.184	**0.662**	**0.211**	**0.275**	**0.229**
		20	0.615	0.148	0.198	0.168	0.649	**0.216**	**0.286**	**0.237**
		30	0.613	0.240	0.213	0.212	0.644	**0.284**	**0.301**	**0.281**
HV	100	10	**0.713**	**0.458**	**0.442**	**0.440**	0.667	**0.316**	**0.325**	**0.310**
		20	**0.686**	0.150	0.226	0.172	**0.686**	0.097	0.220	0.133
		30	**0.711**	0.105	0.250	0.147	**0.711**	0.105	0.250	0.147
	200	10	**0.651**	0.192	0.231	0.206	**0.626**	0.157	0.223	0.180
		20	**0.645**	0.197	0.253	0.220	**0.653**	0.224	**0.258**	0.232
		30	0.670	**0.277**	**0.263**	**0.265**	**0.661**	**0.277**	**0.259**	**0.258**
	300	10	**0.656**	0.162	**0.275**	0.193	0.640	0.120	0.225	0.149
		20	**0.636**	**0.158**	**0.236**	**0.185**	0.649	0.172	0.282	0.208
		30	**0.710**	0.296	0.321	0.300	**0.678**	0.112	0.225	0.148

Overall, the combination of $\omega = 100$ and $k = 10$, coupled with the HV conflict resolution method and $k_{nnc} = 1$, gave the best result; a combination that was also the fastest, 0.49 seconds to classify a single record. The recorded runtime results for all the nine combinations, $\omega = \{100, 200, 300\}$ and $k = \{10, 20, 30\}$, were presented previously in Table 7.3, but are shown in graph form in Figure 7.4 in seconds format. The runtime required by the proposed FMD process, to complete all the nine experiments, was about 22 minutes. The shortest runtime obtained (about half a second) was when the minimum values for the parameters were used, whilst the longest time (about seven seconds) was when the maximum values for the parameters were considered. From Figure 7.4, it can be seen that the runtime increases proportionally as the values associated with the ω and k parameter values increases.

A range of values for λ and σ_m was also experimented with, from 0.005 to 0.100 increasing in steps of 0.005 (thus, twenty values). Both λ and σ_m were considered to be

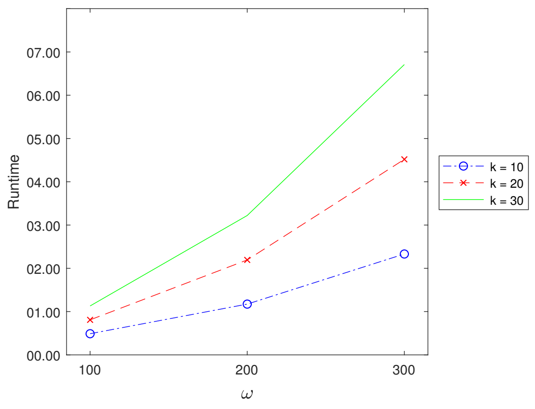


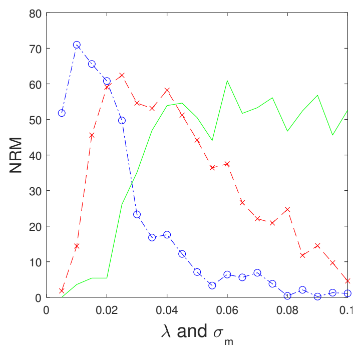
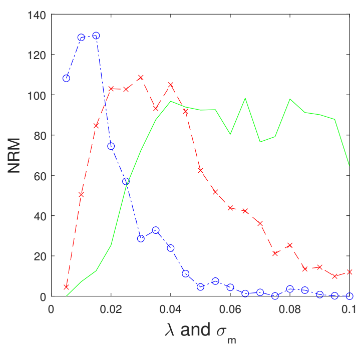
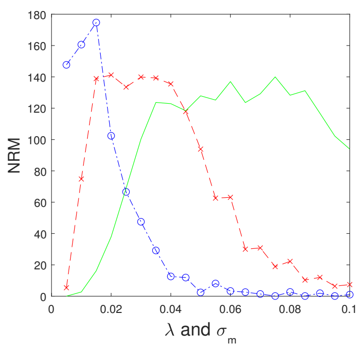
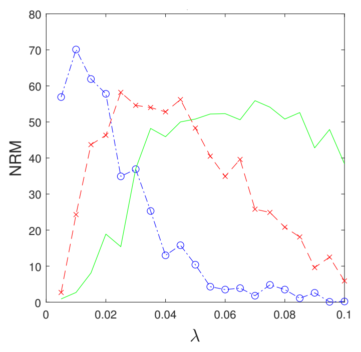
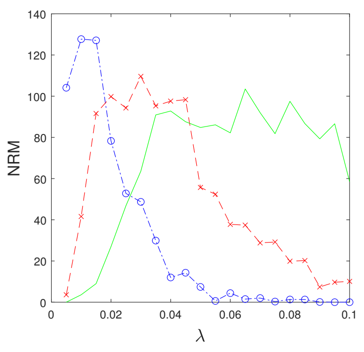
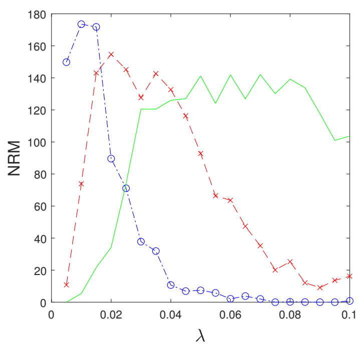
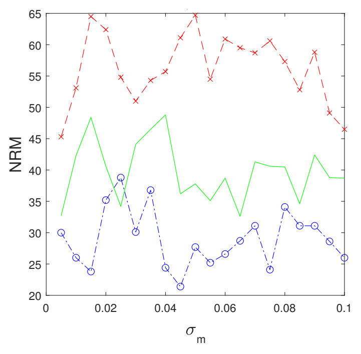
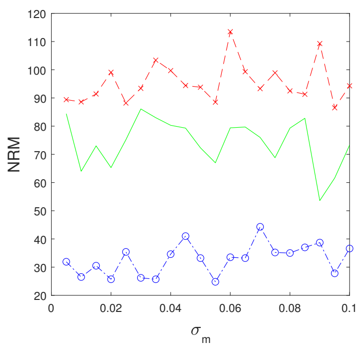
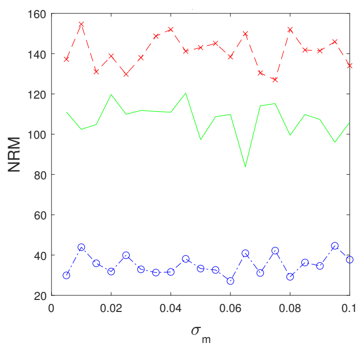
Figure 7.4: Runtime of SGR-FMD approach

more significant with respect to the performance of the proposed approach and thus a greater number of values was considered, compared to the range of values considered for ω, k and max.

The results from the experiments are presented in graph form in Table 7.7. In the table, the columns represent the selected settings for the max parameter. The rows represent the sequence of conducted experiments. The first row presents the results from experiments where the λ and σ_m values were incremented in unison. The second row gives the results where σ_m is held at 0.025 and λ is increased from 0.005 to 0.100 in steps of 0.005; whilst the third row gives the results where λ is held at 0.025 and σ_m is increased from 0.005 to 0.100 in steps of 0.005. The value of 0.025 was used in the second and third sets of experiments because the first set of experiments indicated this to be a most appropriate value. In all cases, the entire data set was used for the experiments. The y-axis of each graph represents the Number of Retained Motifs (NRM) that were derived on completion of the proposed max selection process; note that different scales are used.

Considering the first row of results in Table 7.7, it can first be seen that there are clear "peaks" when $\omega = 100$, while in the case of $\omega = 200$ and $\omega = 300$ a "plateaux" is reached before NRM starts to decrease. As expected, NRM increases as max increases, and tends to decreases as ω decreases. The best general setting for λ and σ_m, regardless of the value of max or ω, can be seen to be in the region of 0.025; thus this was the constant value used for λ and σ_m in all the sets of experiments conducted in this chapter. Considering the second row of results in Table 7.7, it can be seen that similar results were obtained to those reported in the first row. Considering the third row of results, it can be seen that there is no clear best value for σ_m although it is interesting to note that $\omega = 200$ consistently produced best results. This improved performance could be due to the high percentage of pruning when $\omega = 200$ was used. Considering the results given in rows two and three together, it can be concluded that λ has more influence on the criterion, NRM, than σ_m.

Table 7.7: Number of Retained Motifs (NRM) for the SGR-FMD approach using different values for λ, σ_m, max and ω

	max=20	max=40	max=60
λ, σ_m			
λ			
σ_m			

-○- w=100 -×- w=200 — w=300

7.5.4.3 Parameter Settings for the Classification Model

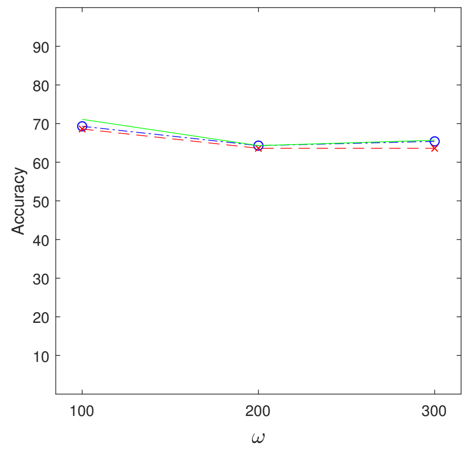
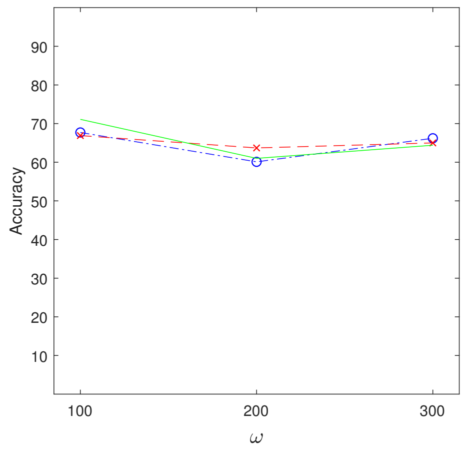
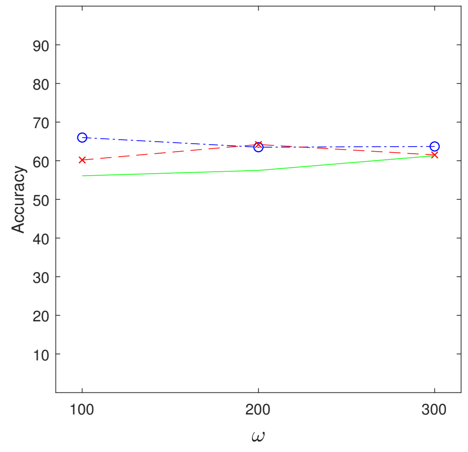
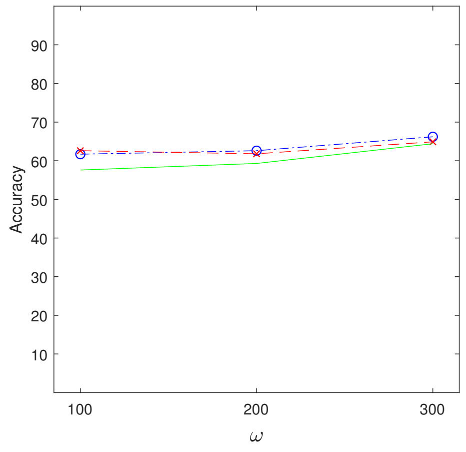
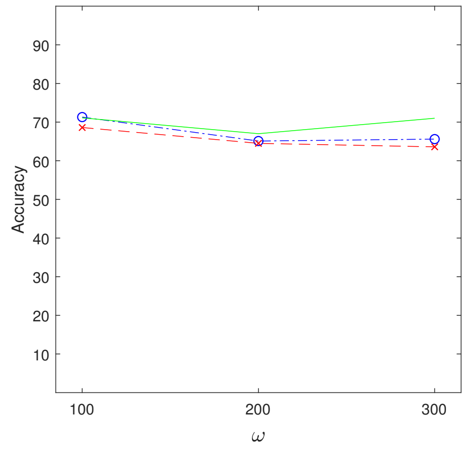
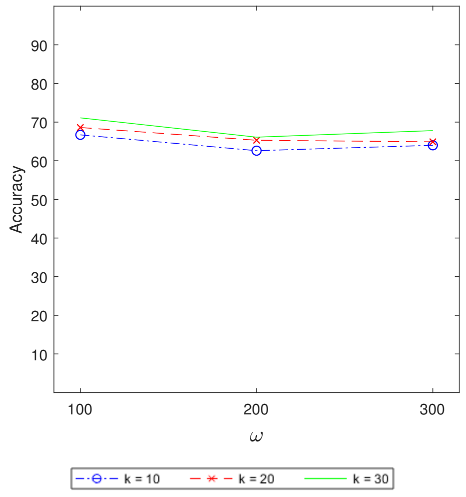
As mentioned in the objectives above, the NNC model was considered with respect to the evaluation presented here. NNC operates by finding the k_{nnc} most similar existing (labelled) records, held in the "motif bank", to a previously unseen record to be classified. The class-labels for the most similar records are then used to label the new record. With respect to the evaluation reported here, similarity was measured using Euclidean Distance (ED) measurement. More specifically, with respect to the evaluation presented here, the bank comprised a set of motifs, $D' = \{\langle m_1, c_1 \rangle, \langle m_2, c_2 \rangle, \dots\}$ where m_i is a frequent motif and c_i is a class-label. The time series to be labelled, the query time series, was then processed, using the proposed SGR-FMD approach, so that it is represented as a set of sub-sequences, $Q' = \{q'_1, q'_2, \dots\}$ each of which is to be matched with the motifs held in D'. In other words, q_i will be labelled based on the k_{nnc} most similar m_i in D'. Two values for the number of nearest neighbours to be identified were used, $k_{nnc} = 1$ and $k_{nnc} = 3$.

Given that the proposed Candidate Frequent Motif Generation process presented in this chapter will typically identify more than one frequent motif in a time series, $|Q'|$ classification labels will be identified; only one is required, the most appropriate label. To select the "winning" class-label, three different Conflict Resolution Methods (CRMs) were considered: Shortest Distance (SD), Shortest Total Distances (STD) and Highest Votes (HV). The SD method simply chooses the class associated with the most similar motif. The STD method chooses the class associated with the lowest accumulated distance; the total similarity distances is calculated for each class and the class with the shortest total distance is selected. The HV method chooses the class with the highest number of votes. In each case, if there is more than one winner, one of the other CRMs is applied.

From the foregoing, a set of two values for k_{nnc}, $\{1, 2\}$, and three CRMs, {SD, STD, HV}, were experimented with to classify unlabelled point series. The metrics used were accuracy, precision, recall and F-score for both $k_{nnc} = 1$ and $k_{nnc} = 3$. For the experiments, $\omega = \{100, 200, 300\}$ and $k = \{10, 20, 30\}$ were again used, coupled with $\sigma_m = 0.025$, $\lambda = 0.025$ and $max = 2 \times k$. The results obtained using the three proposed CRMs, {SD, STD, HV}, were presented in Table 7.6. As can be seen from the table, HV produced the best results in terms of classification accuracy; a best accuracy (in bold) in 14 of the 18 experiments. The SD method provided the second-best results with respect to 7 of the experiments, followed by the STD method which produced only two best accuracy results (in both cases, one of the other methods also produced the best result). In terms of F-score, SD produced the best performance (9 experiments), followed by both STD and HV (5 experiments for each). The best value for k_{nnc} seems to be difficult to identify; although $k_{nnc} = 1$ gave slightly more best accuracy results and produced the best accuracy result in the table (the grey cell).

Although the three considered CRMs were ordered according to their best accuracies, the difference between these accuracies was slight. The accuracy results are therefore presented in graph form in Table 7.8. Inspection of the graphs indicates that accuracy was around 70% for SD and HV, and bellow 70% for STD, regardless of the k_{nnc} value used. It can thus be concluded that there was little to choose between the SD and HV methods.

Table 7.8: Classification accuracy for the SGR-FMD approach using three different conflict resolution methods, and two values for k_{nnc} in classification

	$k_{nnc} = 1$	$k_{nnc} = 3$
SD		
STD		
HV		

7.6 Discussion

Given the results presented in the previous section, it can be concluded that, when using the proposed SGR-FMD approach, the PCG data volume was reduced to between 29% and 36% of its original volume, depending on the ω value used (the motif size). This was reflected in the recorded runtime results, which were reduced significantly; by a factor of more than 278. This runtime advantage is then the principal benefit offered by the SGR-FMD approach, although a small decrease in accuracy was observed. Using the SGR-FMD approach to extract the k most frequent motifs from a PCG time series record required 2.5 seconds on average; whilst in the case of the PCG_{seg} Classification approach (which used a bespoke segmentation technique) an average of 11.6 minutes was require to extract two motifs from a PCG time series record (see Chapter 6).

Looking at the best results obtained by both the SGR-FMD and PCG_{seg} Classification approaches, it would be acceptable if the runtime were to slightly increase (by a matter of seconds) in order to obtain a higher accuracy. As the application of this research is electronic stethoscope diagnosis based on heart sounds, producing a higher accuracy than obtained by the SGR-FMD approach presented in this chapter, at the relatively small expense of a few seconds of extra waiting time, might be worthwhile. Of course, the faster the better, but there is typically a trade-off between accuracy and speed when considering classification methods.

One consideration is that the way of selecting motifs should be precise in order to increase the likelihood of selecting best motifs, and hence improve effectiveness. In the proposed SGR-FMD approach, they were selected from the collection of generated candidate frequent sub-sequences, Q'_i, in a random manner, this could be the cause of the low accuracy. Therefore, it is suggested that motifs should be selected in a more meaningful manner. An alternative mechanism for finding motifs therefore merits consideration, even at the expense of increased runtime. This possibility is explored in the following chapter with respect to another proposed PCG classification approach, the fourth considered in this thesis.

7.7 Summary

This chapter has presented the SGR-FMD approach to PCG classification. The objective of this chapter was to address the challenge of finding discriminative motifs in long time series by proposing two pruning mechanisms: (i) Silent Gap Removal and (ii) Candidate Frequent Motif Generation. The first mechanism PCG_{sgr} was founded on ideas proposed with respect to voice time series to identify words and syllables, the intuition was that little useful information could be extracted from the "silent gaps" between words and syllables. The second mechanism, Candidate Frequent Motif Generation, featured a novel way of clustering sub-sequences, without comparing all sub-sequences with all other sub-sequences, to identify the most frequently occurring sub-sequences. The performance of the proposed SGR-FMD approach was ascertained in the context of runtime and the quality of the motifs identified; runtime was improved by a factor of more than 278; however, a slight drop in effectiveness was observed. The intuition was that the drop in effectiveness could be attributed to the random nature in which motifs were selected using the SGR-FMD approach; an intuition that led to the fourth approach presented in this thesis, the CE-FCS approach. This approach is discussed in further detail in the following chapter.

Chapter 8

The CE-FCS Approach for Phonocardiogram Classification

8.1 Introduction

In the previous chapter, the SGR-FMD approach was presented, applied to Amplitude-PCG time series data. The approach comprised of two core processes: a silent gap removal pre-processing technique (PCG_{sgr}) and a frequent motif discovery algorithm (FMD) for application to the pruned data. At the end of the chapter, it was suggested that, so as to improve the quality of the selected motifs, more "meaningful" motifs should be generated. This is the central theme of this chapter where the Cycle Extraction - Frequent Cycle Selection (CE-FCS) approach to PCG time series classification is presented. The main idea is to pre-process the PCG time series so that "more meaningful" candidate motifs are retained by focusing on the heart cycles that feature in PCG data; the intuition was that these would be more meaningful than those generated using the SGR-FMD approach described in the previous chapter.

As in the case of the SGR-FMD approach, the Amplitude-PCG time series representation was used so that heartbeat cycles could be readily identified, see Figure 2.3(a). This also offered the advantage that the process of detecting cycles can be visualised and checked. The main objective of the work presented in this chapter was thus to generate meaningful motifs, using a cycle extraction technique, so that a useful classification model could be generated in an efficient and effective manner.

The proposed CE-FCS approach comprises two steps:

1. **Cycle Extraction**: A pre-processing technique for the extraction of heartbeat cycles called the PCG_{ce} technique.

2. **Frequent Cycle Selection** (FCS): A two-step process comprised of:

    - **Candidate Frequent Cycle Detection**: Removing of cycles that are infrequent, using a novel technique involving the distribution of cycles.

- **Frequent Motif Discovery**: Discounting cycles that are considered not to be good discriminators of a single class.

As in the case of the previously presented PCG classification approaches, the relevant symbols with respect to the CE-FCS technique presented in this chapter are given in Table 8.1.

The remainder of this chapter is organised as follows. The Cycle Extraction - Frequent Cycle Selection approach is presented in Section 8.2. Section 8.3 provides a description of the proposed PCG_{ce} cycle extraction pre-process. The proposed FCS process is then presented in Section 8.4, which includes descriptions of the Candidate Frequent Cycle Detection and Frequent Motif Discovery sub-processes. In Section 8.5, an extensive analysis of the results obtained from experiments conducted to evaluate the proposed CE-FCS approach is presented. Some further discussion is given in Section 8.6, and finally in Section 8.7, some conclusions concerning the information presented in this chapter are provided.

8.2 The CE-FCS Approach

This section presents the proposed Cycle Extraction - Frequent Cycle Selection (CE-FCS) PCG classification approach. A schematic of the proposed approach is given in Figure 8.1. The figure shows the "Cycle Extraction" pre-process and the two sub-processes, "Candidate Frequent Cycle Detection" and "Frequent Motif Discovery", within the FCS process. The example training data set, in the figure, is the same as that used for the schematics given in Chapters 5 to 7; nine labelled time series: 4, 3 and 2 time series belonging to the classes blue, green and red respectively. It should be noted that, as demonstrated by the evaluation results presented later in this chapter, that the number of motifs of any class retained at the end of the process (in the "motif bank") does not seem to be unduly affected by the number of the time series associated with an individual class in the training data set. Thus, in the example shown in the figure, the blue class features the highest number of time series but, by the end of the process, results in the lowest number of motifs. From the figure, it should also be noted (as in the case of the schematics given in Chapters 5 to 7) that the input to processes indicated by multiple arrows are intended to indicate that time series could be processed in parallel; whereas a single arrow indicates that parallel processing is not possible.

The CE-FCS approach starts with the Cycle Extraction process where the proposed PCG_{ce} technique is applied. This is the data pre-processing step which is discussed in detail in the following section, Section 8.3. The extracted cycles are then sent to the Candidate Frequent Cycle Detection sub-process. As can be seen from the figure, the cycles associated with each class are processed in turn.

During the Candidate Frequent Cycle Detection process, the distance, d, between each cycle and a "zero-motif" is calculated (the zero-motif concept was discussed previously

Table 8.1: CE-FCS approach symbol table

p	A numeric value representing a point in a point (time) series.
P	A point (time) series comprising a sequence of points $\{p_1, p_2, \dots\}$ and, in the case of PCGs, consisting of a number of cardiac cycles.
C	A set of classes $\{c_1, c_2, \dots\}$.
h	A heartbeat, a sub-sequence of P considered to form a cardiac cycle, $h \subset P$.
m	A motif, a heartbeat h that is deemed (in some sense) to be representative of a class.
zm	The zero-motif, a hypothetical motif comprised of only zero values.
T	A set of point series and class-label pairs $\{\langle P_1, c_1\rangle, \langle P_2, c_2\rangle, \dots\}$.
H	A set of cycle and class-label pairs $\{\langle h_1, c_1\rangle, \langle h_2, c_2\rangle, \dots\}$.
ζ	The z-score, in a Gaussian Distribution, used with respect to the distribution of D_z.
z_1,z_2	Two numbers forming the boundaries of ζ.
H'	The set of pairs in H which appear within the range of z_1 and z_2, $H' \subset H$.
$\overline{H}$	A set of cycles of a certain class-label c_i, $\overline{H} \subset H$.
H''	The set of pairs in H' that are good class-label discriminators, $H'' \subset H'$.
D_z	A set holding similarity values $\{d_{z_1}, d_{z_2}, \dots\}$, where d_{z_i} corresponds to $\overline{h_i} \in \overline{H}$.
f	The frequency with which a cycle $\overline{h}$ occurs in $\overline{H}$.
M	A set of motif and frequency value pairs $\{\langle m_1, f_1\rangle, \langle m_2, f_2\rangle, \dots\}$.
E	The energy of P comprising a sequence of values $\{e_1, e_2, \dots\}$.
E'	The standardised E.
V	The envelope of E' consisting of a number of oscillations.
X	A set of points in V that might lead to the boundaries of cardiac cycles h_i in P.
X'	A set of points in V that are anticipated to form the boundaries of h_i in P.
μ_d	The mean of D_z.
μ_e	The mean of E.
σ_d	The standard deviation of D_z.
σ_e	The standard deviation of E.
σ_m	A pre-specified frequency threshold.
α	A pre-specified oscillation-width threshold.
λ	A pre-specified similarity threshold for comparing two sub-sequences or motifs.
max	A pre-specified maximum size for a set of motifs M.
$flag$	A pre-specified decision for eliminating the size of each class in M to be either max or not (all).
k	A pre-specified threshold limiting the number of motifs in M to the k most frequently occurring motifs, $k < max$.
t	A dynamically computed threshold for detecting cycles h_i in V.

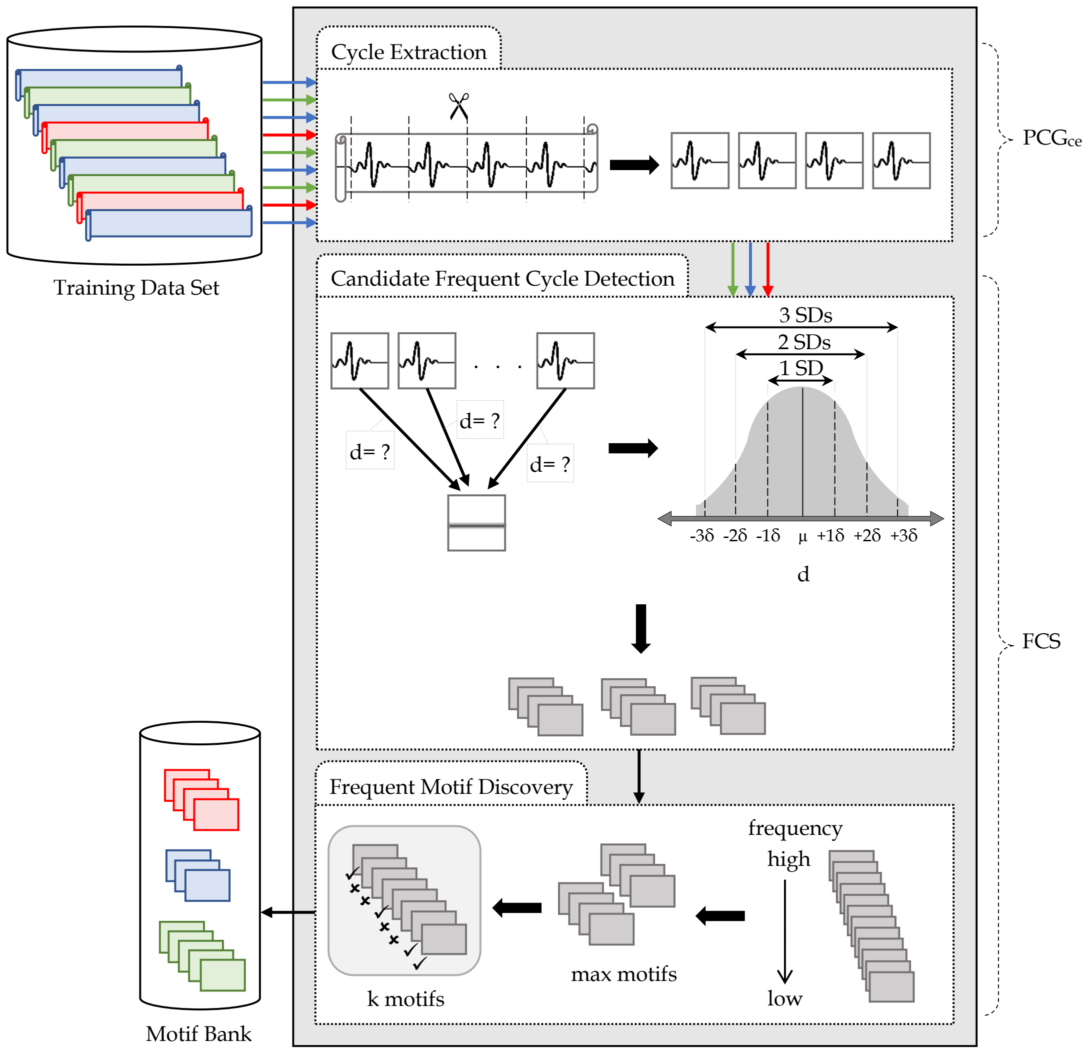


Figure 8.1: Schematic showing the basic operations of the CE-FCS approach

in Chapter 7). After all the distance values for all cycles in the given class have been calculated, their Gaussian distribution is considered. Cycles whose associated distance value falls within a given number of standard deviations from the mean (μ_d) distance value, defined by the threshold ζ, are retained. The process is repeated for each class and the collected candidate frequent cycles passed to the Frequent Motif Discovery sub-process.

The Frequent Motif Discovery process commences by determining the "real" distances between the cycles (as opposed to the computationally less expensive zero-motif distance). The cycles are then arranged in descending order according to their frequency so that the most frequent can be selected. At this stage, one option was to consider pruning the retained cycles so that only a predefined number, *max*, were retained; the alternative was to retain all cycles. Whatever the case, the next step was to group the retained cycles according to their class-label. Each cycle is then compared to each other cycle in each other group. The cycles that are found to be similar to cycles in other groups are marked with an "**x**" (not good discriminators of class). The remaining cycles are then processed further. The algorithm needs only the top k frequent cycles per class, if available. These are marked with a "✓" (good discriminators of class) and the process is terminated for that class. In the example given in Figure 8.1 $k = 5$ was used, the example motif bank however contains: 3, 4 and 5 motifs belong to the classes blue, red and green respectively because k is a maximum. The identified cycles then form the "motif bank" to be used to label previously unseen time series.

The pseudo code for the CE-FCS approach parent process is given in Algorithm 20. The inputs are: (i) a set T of point series and class pairs, (ii) a set C of class-labels, (iii) the maximum size max for specifying the number of cycles in M, (iv) a threshold k for limiting the number of frequent motifs selected, (v) a threshold α for extracting cycles, (vi) a threshold σ_m for defining the concept of frequency, (vii) a threshold λ for defining the concept of similarity, (viii) a distribution threshold ζ used for determining which cycles to retain and (ix) a pruning flag, $flag$, set to "max" if M is to be pruned, and "all" otherwise. The output is a set H'' of frequently occurring motifs that are considered to be good discriminators of class.

Algorithm 20 CE-FCS Approach

```
Require: $T$, $C$, $max$, $k$, $\alpha$, $\sigma_m$, $\lambda$, $\zeta$, $flag$
Ensure: $H''$
 1: $H \leftarrow \emptyset$, A set to hold cycle-class pairs
 2: $H' \leftarrow \emptyset$, A set to hold subset of $H$, the frequent pairs
 3: for $\forall \langle P_i, c_i \rangle \in T$ do
 4:     $H_i \leftarrow PCG_{ce}(P_i, c_i, \alpha)$
 5:     $H \leftarrow H \cup H_i$
 6: end for
 7: for $\forall c_i \in C$ do
 8:     $\overline{H}_i \leftarrow$ A set of cycles of class $c_i$ deducted from $H$
 9:     $H'_i \leftarrow candidateFreqCycDetection(\overline{H}_i, c_i, \zeta)$
10:     $H' \leftarrow H' \cup H'_i$
11: end for
12: if $H' \neq \emptyset$ then
13:     $H'' \leftarrow freqMotifDiscovery(H', C, max, k, \sigma_m, \lambda, flag)$
14: end if
15: return( $H''$ )
```

The algorithm commences by initialising the sets H and H' with $\emptyset$ (empty set). Each time series and class pair, $\langle P_i, c_i \rangle$ in T, is processed in turn. The cycles in each time series P_i are then extracted using the PCG$_{\text{ce}}$ process described later in Section 8.3 (lines 3 to 6). The resulting cycle and class pairs, $\langle h_i, c_i \rangle$, are stored in H. Each class is then processed in turn (lines 7 to 11) and the size of the set H is reduced by removing infrequent cycles so as to give a set of cycles H'. The cycles associated with the class c_i are collected in a set $\overline{H}_i$ (line 8). This is then processed (line 9), using the Candidate Frequent Cycled Detection sub-process (*candidateFreCycDetecion*) described in Sub-section 8.4.1 below, to identify the set of frequent cycles H'_i. All frequent cycles from all classes are stored in the set H'

(line 10). This is then further processed (line 13), using the Frequent Motif Discovery process (*freqMotifDiscovery*) described in Sub-section 8.4.2, to identify the set H'', the set of k most frequent cycles which can be then used as a "motif bank" in a Nearest Neighbour Classification (NNC) model to label previously unseen PCG time series.

From Algorithm 20, it can be seen that there are three function calls. The first call (line 4) is for the pre-processing technique, PCG_{ce}, presented in the following section, Section 8.3. The remaining two function calls (lines 9 and 13) are for the two sub-processes making up the FCS parent process discussed in further detail in Section 8.4.

8.3 Cycle Extraction

This section presents the PCG_{ce} cycle extraction technique. A number of pre-processing methods were presented in Chapter 2, where it was noted that, in the case of Amplitude-PCG data, it is possible to extract the heartbeat cycles and then to isolate the cycle components. PCG signals comprise cycles, each comprised of: (i) a heartbeat, (ii) some murmurs and clicks, if diseased, and (iii) noise, if applicable. In a cycle there are two heart sound components, S1 and S2, corresponding to the closing of the atrioventricular valve and then the semilunar valve. A cycle is measured, in this chapter, from the start of the S1 component to the start of the following S1 component. The central idea underpinning the CE-FCS approach presented in this chapter was to segment a data set of labelled PCG signals into a collection of cycles with cycles grouped (clustered) according to class-label. This idea is common in the field of Signal Processing and has been applied previously to PCG signals, for example to study the duration of S1 or to find what are known as "click positions" [110, 159, 177]. The proposed technique in this section differs from this previous work in that the focus is on "whole cycles" rather than their components. The technique, was founded on that presented in [58], but with modifications, and operates using a dynamic threshold computed for each signal (PCG time series) to detect the beginning of cycles.

For each point series $P = \{p_1, p_2, \dots\}$, two parameters are calculated: (i) the standardised signal energy envelope, V, and (ii) the threshold, t. The first, V, is the signal (time series) energy E which is standardised to give E' and then the encompassing "envelope" extracted. The threshold t, as will become clear later in this chapter, is used to define an Amplitude "cut-off" value used to detect the beginning of cycles. The energy $E = \{e_1, e_2, \dots\}$ is computed by squaring the Amplitude values (p_i) in P, where each e_i in E is calculated as shown in Equation 8.1:

$$e_i = p_i^2 \tag{8.1}$$

There are many other ways to calculate the energy of a signal, such as using absolute value, Shannon entropy or Shannon energy [58]; each of them serves a goal. The aim in the context of the work presented in this chapter is to detect the beginning of cycles, the start of the S1 component, usually the component with the highest Amplitude (the loudest) [40].

The above method of calculating the energy (squaring the points) was therefore adopted because point values with high Amplitude will be favoured over those with low Amplitude. This will in turn facilitate S1 detection.

The standardised energy $E' = \{e'_1, e'_2, \dots\}$ is then calculated as presented in Equation 8.2:

$$e'_i = \frac{e_i - \mu_e}{\sigma_e} \tag{8.2}$$

Where μ_e and σ_e are the mean and standard deviation of E respectively. The envelope V of E' can thus be defined and used to detect the beginning of cycles using the Amplitude "cut-off" value t.

The process is illustrated in Figure 8.2. The process starts by identifying the oscillation in V with the highest energy value, the magenta oscillation in Figure 8.2(a). Then, the predefined α threshold, a percentage of the width of the oscillation with the highest Amplitude (the start and end of an oscillation can be identified from trend changes in V), is used to determine the value for t, the cyan line shown in Figure 8.2(b) (and Figure 8.2(c)). The value for t is used to find ascending intersection points in V as shown in Figure 8.2(c). Any oscillations in the energy envelope V whose Amplitude falls below t are ignored, because they are deemed to be clicks and murmurs. Using this process, some ascending intersection points demarcating S2 components will still be retained, as illustrated in Figure 8.2(c). To remove these, the distance between intersection points is considered, if this falls below the average distance then we have an S2 intersection point which should be ignored. The retained points are then used to "track" back along the envelope until a change in trend is discovered; this marks the start of an S1 component and thus the start of a cycle, the cycle ends with the start of the following S1 component.

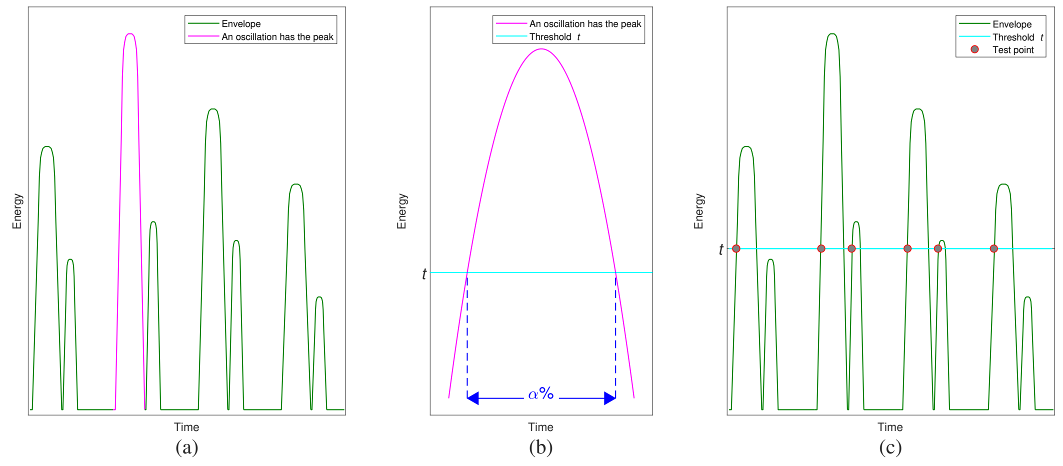


Figure 8.2: Dynamic t value calculation using a PCG envelope signal: (a) example PCG envelope with the highest Amplitude oscillation highlighted, (b) t value calculation and (c) intersect points

The pseudo code for the PCG_{ce} technique, incorporating the above, is given in Algorithm 21. The inputs are: (i) a point series P, (ii) a class-label c and (iii) an oscillation-width threshold α. The output is a set H containing identified cycle and class pairs,

$H = \{\langle h_1, c\rangle, \langle h_2, c\rangle, \dots\}$. The technique commences by initialising the sets H and X' with $\emptyset$ (empty set), and defining two variables: $startS1$ to hold the start index of an oscillation in an envelop V and avg to hold the average of the distances between intersection points. The process then operates by calculating the signal energy E (line 5) and then standardising this (line 6). The energy envelope V is then determined (line 7). Next, a value for t is calculated using the threshold α (line 8). This is then used to populated a set X of intersection points (line 9) from which a set $\overline{X}$ of distances between points in X is calculated and a consequent average distance value derived (lines 10 and 11). The set $\overline{X}$ is then processed (lines 12 to 17) and points in X associated with distances below the average removed from X; in other words, intersection points indicating S2 components are removed. The remaining points in X, indicating locations in V, are used to track back along the energy envelope to identify cycle start points (change in trend points); these are stored in X'. Consecutive cycle start points in X' are used to define cycles in P. Each index value in X' is processed in turn (lines 18 to 23), except the last value, to define a set of cycles each of which starts with a point x'_i and ends just before the next start point x'_{i+1}; these cycles are stored in H together with the given class-label, c. The algorithm exits with H (line 24).

Algorithm 21 PCG$_{\mathrm{ce}}$

```
Require: $P$, $c$, $\alpha$
Ensure: $H$
1: $H \leftarrow \emptyset$, A set to hold cycle-class pairs
2: $X' \leftarrow \emptyset$, A set to hold points expected to be the beginning of cycles
3: $startS1$, A variable to hold a point value marking the start of S1 component
4: $avg$, A variable to hold an average value
5: $E \leftarrow$ Signal energy for $P$ calculated using Equation 8.1
6: $E' \leftarrow$ Standardisation of signal energy $E$ calculated using Equation 8.2
7: $V \leftarrow$ Envelope of $E'$
8: $t \leftarrow$ A “cut-off” value computed for $V$ using the threshold $\alpha$ as in Figure 8.2(b)
9: $X \leftarrow$ A list of test points extracted from $V$ as in Figure 8.2(c)
10: $\overline{X} \leftarrow$ A list of distances between points in $X$
11: $avg \leftarrow$ Average distances in $\overline{X}$
12: for $\forall \overline{x_i} \in \overline{X}$ do
13:    if $\overline{x_i} > avg$ then
14:       $startS1 \leftarrow$ Point in $V$, tracking back from $x_i$, marking the start of increase
15:       $X' \leftarrow X' \cup startS1$
16:    end if
17: end for
18: for $\forall x'_i \in X'$ do
19:    if $i \neq |X'|$ then
20:       $h \leftarrow \forall\, p_j \in P : x'_i \leq j < x'_{i+1}$
21:       $H \leftarrow H \cup \langle h, c\rangle$
22:    end if
23: end for
24: return( $H$ )
```

8.4 Frequent Cycle Selection

In Section 8.2, a high-level overview of the proposed CE-FCS approach was presented. The approach comprises two processes, Cycle Extraction (PCG_{ce}) and Frequent Cycle Selection (FCS). The first was discussed in the above section, the second is discussed in this section.

As discussed earlier in Chapter 2, a motif is a time series sub-sequence that is representative of a class which can be used for the purpose of time series classification. Determining whether a motif is representative of a time series or not is explored, in the context of this thesis, using two strategies: (i) similarity- and (ii) frequency-based. With respect to the motif discovery algorithm used in the first two approaches presented in this thesis, the MK Benchmark and the PCG_{seg} Classification approaches (discussed in Chapters 5 and 6), the similarity-based strategy was used because this is what was proposed in [112] with respect to the original MK algorithm. The alternative strategy to extract motifs based on frequency, was explored in the previous chapter. The advantage, demonstrated in Chapter 7, was that more representative motifs could be discovered leading to more accurate PCG classification. The trade-off was greater time complexity, which was mitigated against in the previous chapter using the concept of silent gap removal and in this chapter using the concept of cycle extraction.

For the proposed motif discovery algorithm with respect to this chapter, FCS, representativeness was measured from the frequency-based perspective. Given a set $\overline{H}$ of cycles associated with a particular class, a frequent motif is a cycle $\overline{h} \in \overline{H}$ that occurs in $\overline{H}$ more often than some threshold σ_m. In other words, for $\overline{h_j}$ to be considered to be a frequent motif, the set $\overline{H}$ must include at least σ_m cycles that are in some sense all similar to $\overline{h_j}$, as defined according to a similarity threshold λ. The set of candidate frequent motifs generated from $\overline{H}$ is then given by $M = \{\langle m_1, f_1\rangle, \langle m_2, f_2\rangle, \dots\}$, where f_i is the frequency count ($f_i \geq \sigma_m\%$ of $|\overline{H}|$).

Given a set of candidate frequent motifs M, the k most frequently occurring motifs associated with a class-label c_i are, arguably, the most representative of their class c_i. The top k motifs drawn from the set of frequent motifs M is given by the set $H'' = \{\langle h''_1, c_1\rangle, \langle h''_2, c_2\rangle, \dots\}$. The top k motifs should not be similar to other motifs in H'' that are associated with different class-labels. In other words, a motif in the set H'' is defined as one where either there are no similar motifs in H'' or, if there are similar motifs in H'', they are all linked to the same class; H'' can be more formally defined as follows:

- $H'' = \{\langle h''_1, c_1\rangle, \langle h''_2, c_1\rangle, \dots, \langle h''_k, c_1\rangle,$
  $\langle h''_{k+1}, c_2\rangle, \langle h''_{k+2}, c_2\rangle, \dots, \langle h''_{2\times k}, c_2\rangle,$
  $\dots,$
  $\langle h''_{(|C|-1)\times k+1}, c_{|C|}\rangle, \langle h''_{(|C|-1)\times k+2}, c_{|C|}\rangle, \dots, \langle h''_{|C|\times k}, c_{|C|}\rangle\}.$

- $\forall\ \langle h''_i, c_i\rangle \in H'' \nexists\ \langle h''_j, c_j\rangle \in H''\ :\ h''_i \simeq h''_j\ \wedge\ c_i \neq c_j.$

As mentioned above, the proposed motif discovery algorithm, FCS, comprises two sub-processes, Candidate Frequent Cycle Detection and Frequent Motif Discovery; each of which is detailed in the following two sub-sections, Sub-sections 8.4.1 and 8.4.2.

8.4.1 Candidate Frequent Cycle Detection

As in the case of the Frequent Motif Discovery (FMD) algorithm described in the previous chapter, the idea is to prune, early on in the process, cycles (sub-sequences) that cannot be frequent in order to enhance the efficiency of the proposed FCS process, and consequently the CE-FCS approach. This pruning leads to retaining only a set of candidate frequent cycles, H', that are good indicators of a class; H' is thus a subset of H. Recall that the intuition was that frequent cycles would equate to motifs. To find cycle frequency, a novel algorithm was proposed in Chapter 7 which used a hypothetical motif (cycle in this chapter) zm, the "zero-motif", holding only zero values, $zm = \{zm_j : zm_j = 0,\ 1 \leq j \leq |h| \ \forall\ h \in H\}$. The similarity between each heartbeat $h_i \in H$, and zm was calculated using an Euclidean Distance similarity function, $d(h_i, zm)$. However, given that zm is a vector of zeros, the similarity function can be simplified as presented in Equation 8.3, which coincides with Root Sum Square (RSS) calculation.

$$d(h_i, zm) = \sqrt{\sum_{j=1}^{|h_i|} h_{i_j}^2} \tag{8.3}$$

Since the length of each cycle $|h_i|$ is not fixed, the similarity value was normalised by dividing it by the cycle length; the similarity function thus becomes as shown in Equation 8.4. This was then the function used to populate the set of distances D_z.

$$d(h_i, zm) = d_{z_i} = \frac{\sqrt{\sum_{j=1}^{|h_i|} h_{i_j}^2}}{|h_i|} \tag{8.4}$$

RSS assumes that most of the obtained values fall to the middle of the value range and describe a Gaussian distribution. The zero-motif distances were arranged in bins (the x-axis of a Gaussian distribution curve), for which the mean (μ_d) and standard deviation (σ_d) values were calculated. Using μ_d, σ_d and a given number of standard deviations threshold ζ; the value of two variables z_1 and z_2 can be calculated using Equation 8.5. The variables z_1 and z_2 are designed to limit the range of frequent similarity values; the cycles associated with the bins falling within the range defined by z_1 and z_2 are retained to give H'.

$$z_1 = \mu_d + (\zeta \times \sigma_d) \qquad z_2 = \mu_d - (\zeta \times \sigma_d) \tag{8.5}$$

The pseudo code for the Candidate Frequent Cycle Detection sub-process is given in Algorithm 22. The inputs are: (i) a set of cycles, $\overline{H}$, of a given class, (ii) their class-label c, and (iii) a threshold ζ. The output is a set of candidate frequent cycles H'. The algorithm commences (line 1) by defining the zero-motif zm, this is included for clarity purposes, in practice zm does not need to be specifically defined. Then, the sets H' is initialised with $\emptyset$ (the empty set). An empty set D_z corresponding to the given set $\overline{H}$ is also initialised with $\emptyset$ to hold the distance (similarity) values from zm, once calculated. First, the set $\overline{H}$ is processed (lines 4 to 7) so as to populate D_z; note that there is a one-to-one correspondence between $\overline{H}$ and D_z. The μ_d and σ_d parameters of the set D_z are then extracted (lines 8 and 9). Given the calculated μ_d and σ_d values and the ζ input parameter, the limits z_1 and z_2 are calculated (line 10). The cycles in $\overline{H}$ associated with distances (bins) within the range between z_1 and z_2 are stored in H' (lines 11 to 15). The retained set of cycles H' is then returned (line 16).

Algorithm 22 Candidate Frequent Cycle Detection

```
Require: $\overline{H}$, c, $\zeta$
Ensure: $H'$
 1: $zm \leftarrow$ The zero-motif
 2: $H' \leftarrow \emptyset$, A set to hold frequent cycles, which are within $\zeta$ standard deviations
 3: $D_z \leftarrow \emptyset$, A set to hold distance values
 4: for $\forall \overline{h_i} \in \overline{H}$ do
 5:     $d_{z_i} \leftarrow$ Euclidean similarity between $\overline{h_i}$ and $zm$ calculated using Equation 8.4
 6:     $D_z \leftarrow D_z \cup d_{z_i}$
 7: end for
 8: $\mu_d \leftarrow$ The mean of the set $D_z$
 9: $\sigma_d \leftarrow$ The standard deviation of the set $D_z$
10: $z_1$, $z_2 \leftarrow$ Two numbers calculated using $\mu_d$, $\sigma_d$, $\zeta$ and Equation 8.5
11: for $\forall \overline{h_i} \in \overline{H}$ do
12:     if $z_2 \leq d_{z_i} \leq z_1$ then
13:         $H' \leftarrow H' \cup \langle \overline{h_i}, c \rangle$
14:     end if
15: end for
16: return( $H'$ )
```

8.4.2 Frequent Motif Discovery

The final process in the FCS algorithm is the Frequent Motif Discovery sub-process. The aim of this sub-process, given a set of candidate frequent cycles H' generated using the Candidate Frequent Cycle Detection sub-process described in the foregoing sub-section, is to identify the most frequent motifs (cycles) that, by definition, are deemed to be good discriminators of class. However, the number of remaining cycles in H' may still be large and some may not necessarily be good discriminators of class. As in the case of the SGR-FMD approach described in the previous chapter, an optional step is therefore to limit the number of candidate frequent cycles to be considered using a user-defined threshold max. Thus, the idea is to select the max most frequent candidates from the set H'. If this option is not chosen, all the frequent candidates in the set H' will be considered.

The pseudo code for the Frequent Motif Discovery algorithm is presented in Algorithm 23. The inputs are: (i) a set of candidate frequent cycles H'; (ii) a set C of all class-labels; (iii) the thresholds σ_m, λ, k and max; and (iv) a pruning decision, $flag$, set to "max" if M is to be pruned, and "all" otherwise. The output is a set H'' of identified frequent cycles.

Algorithm 23 Frequent Motif Discovery

```
Require: $H'$, $C$, $max$, $k$, $\sigma_m$, $\lambda$, $flag$
Ensure: $H''$
 1: $H'' \leftarrow \emptyset$, An empty set to hold top $k$ frequent cycles
 2: $M \leftarrow \emptyset$, An empty set to hold frequent cycle and frequency pairs
 3: for $\forall c_i \in C$ do
 4:     $\overline{H} \leftarrow$ A set of cycles of class $c_i$ deducted from $H'$
 5:     for $\forall \overline{h_j} \in \overline{H}$ do
 6:         $f_j \leftarrow$ The number of cycles in $\overline{H}$ that are similar to $\overline{h_j}$ according to the threshold $\lambda$
 7:         if $f_j \geq \frac{\sigma_m \times |\overline{H}|}{100}$ then
 8:             $M \leftarrow M \cup \langle \overline{h_j}, f_j \rangle$
 9:         end if
10:     end for
11:     if $M \neq \emptyset$ then
12:         Reorder the set $M$ according to frequency values in descending order
13:         if $flag =$ "max" and $|M| > max$ then
14:             $M \leftarrow$ A set of first $max$ motifs (cycles) in $M$
15:         end if
16:         $count \leftarrow 0$, A counter for selecting $k$ cycles from $M$
17:         for $\forall m_j \in M$ and $count < k$ do
18:             if $isGoodDiscriminator(H', m_j, c_i)$ then
19:                 $H'' \leftarrow H'' \cup \langle m_j, c_i \rangle$
20:                 $count \leftarrow count + 1$
21:             end if
22:         end for
23:     end if
24: end for
25: return( $H''$ )
```

The algorithm commences by initialising the sets H'' and M with $\emptyset$ (the empty set). The algorithm processes the cycles associated with each class separately (lines 3 to 24). In each iteration, the generated set $\overline{H}$ is comprised of the complete set of cycles associated with the class-label c_i and the content deducted from H' (line 4). Next, the frequency count f_i for each cycle $\overline{h_j} \in \overline{H}$ is calculated (lines 5 to 10) using Euclidean Distance and the λ parameter for determining the similarity between cycles; note that the similarity comparisons are abandoned as soon as the similarity value exceeded λ. In each case, if the count is greater than or equal to σ_m% of $|\overline{H}|$, the cycle and its frequency value are stored in a set $M = \{\langle m_1, f_1\rangle, \langle m_2, f_2\rangle, \dots\}$. Note that if a high σ_m threshold value is used, the set M may be empty, this is tested for in line 11. If the set M is not empty, it is reordered, in descending order of frequency value (line 12). The next step (lines 13 to 15) depends on the pruning decision flag, if $flag$ = "max", only the max most frequent cycles are retained in the set M. Otherwise, the algorithm goes directly to line 16 where a counter, *count*, is set for selecting the k most "discriminative" motifs from the set M. The most discriminative motifs are considered to be the most frequently occurring cycles that are associated with only one class (there are no similar cycles associated with other classes). The function *isGoodDiscriminator* (line 18) therefore ensures that by comparing each cycle m_j in M with all other cycles in H' that are associated with different class-labels; only the discriminative cycles are added to the set H'' (line 19). Similarity is measured in the same way as before using Euclidean Distance and the λ parameter.

8.5 Evaluation

In the foregoing, the operation of the Cycle Extraction technique (PCG_{ce}), and Frequent Cycle Selection algorithm (FCS) were presented. These two processes form the CE-FCS approach, the evaluation of which is reported on in this section. As in the case of the evaluation reported in the three previous chapters, the evaluation was conducted using the collected evaluation data set described in Chapter 3. The objectives of the evaluation were:

Objective 1, Operational Analysis: To investigate the operation of the proposed approach.

Objective 2, Pruning Techniques Analysis: To investigate the effect of the pruning techniques used on the data size.

Objective 3, Comparison with the SGR-FMD Approach: To compare the operation of the proposed CE-FCS approach with the SGR-FMD approach in terms of computational efficiency (runtime) and effectiveness (accuracy).

Objective 4, Most Appropriate Parameter Settings: To establish the most appropriate parameter settings for:

- The Cycle Extraction pre-processing technique (PCG_{ce}), which uses the parameter α, the oscillation width.
- The Frequent Cycle Selection (FCS) process, which uses the parameters: (i) a similarity threshold λ, (ii) a frequency threshold σ_m, (iii) a number of standard deviations threshold ζ, (iv) two selection thresholds k and max, and (v) the pruning decision option $flag$.
- The adopted NNC classification model, and the most appropriate method for conflict resolution, where a number of motifs is considered.

Each of the above objectives are considered in turn in the following four sub-sections, Sub-sections 8.5.1 to 8.5.4. The evaluation metrics recorded were accuracy (acc), precision (prec), recall (rec), F-score (f-s) and runtime. Five-fold cross-validation was adopted and the data was statically stratified with respect to all the reported experiments. The evaluation was conducted using the Java programming language. The experiments were run on an iMac Pro (2017) computer with 8-Cores, 3.2GHz Intel Xeon W CPU and 19MB RAM.

8.5.1 Operational Analysis

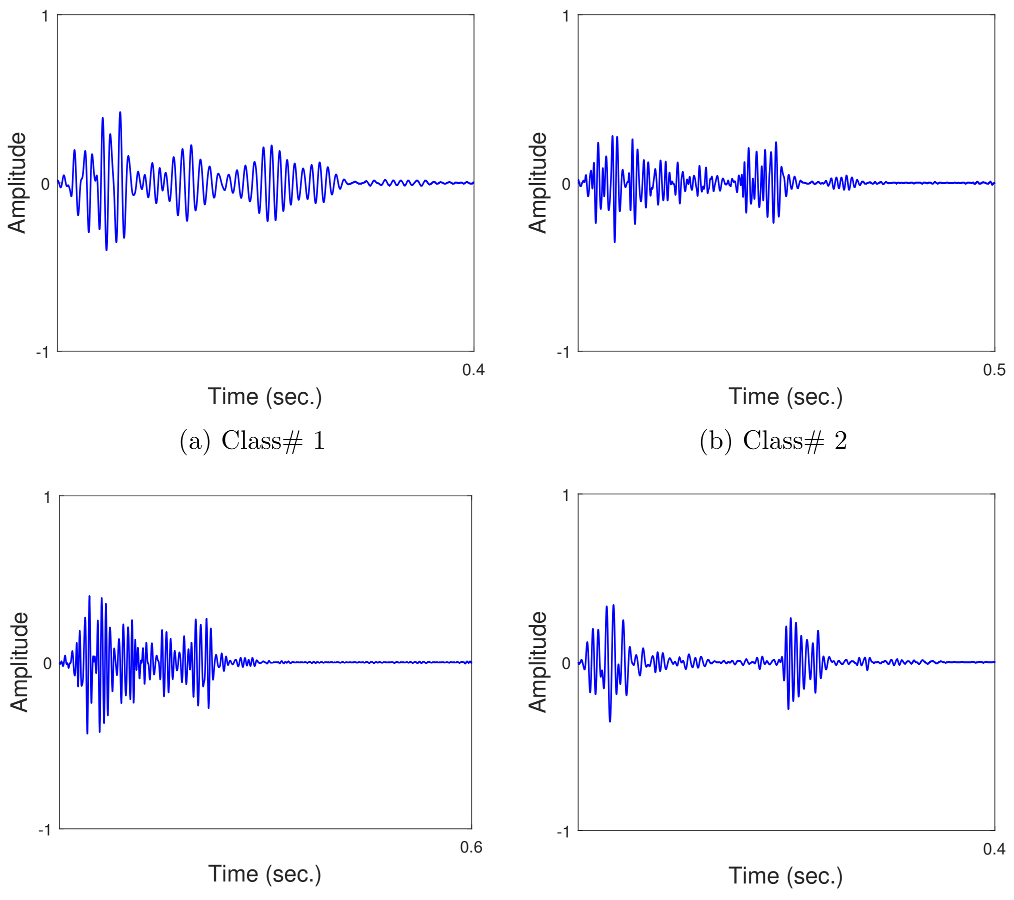
In this sub-section, the evaluation of the operation of the proposed CE-FCS approach is presented. Examples of cycle motifs for each of the four classes obtained during the evaluation of the approach are given in Figure 8.3. As the figure shows, each motif is a complete cycle. Using NNC and five-fold cross-validation, the best and worst accuracies recorded were 72.0% and 53.3%. The precisions, recalls and F-scores were 0.253, 0.311 and 0.272 for the best accuracy, and 0.042, 0.076 and 0.041 for the worst. In addition, the recorded standard deviation for the classification model was good, giving an average of around 0.05. The average runtime to extract the k most frequent motifs from one record was 0.65 seconds. The total time for the whole data set was less than a minute (38.45 seconds), on average, for each experiment. A set of 90 experiments was conducted to determine the best values for the FCS algorithm variables, more details concerning these experiments are given in Sub-section 8.5.4.

8.5.2 Pruning Techniques Analysis

In the proposed CE-FCS approach, two pruning techniques were used: (i) extracting cycles using PCG_{ce} and (ii) identifying candidate frequent cycles. Applying the former technique, the input time series P was reduced by about 11%. The Candidate Frequent Cycle Detection then takes as input a set of cycles H (the output from the PCG_{ce} technique) and a threshold ζ defining the number of standard deviations to select the most frequent cycles and prune the remainder. The proposed method involved determining the Gaussian distribution of the similarity values (distances) of the given cycles and then selecting those that were within ζ standard deviations.

A histogram of the distances of each class in the evaluation data set was plotted where the number of bins was a tenth of the number of distances to be distributed. This is illustrated in Figure 8.4 which shows the distribution for each class in the evaluation data set (the red line is the best-fit curve). From the figure, it can be seen that a Gaussian distribution curve results with long tails for all four classes. This tail was trimmed because the focus of the proposed method is the frequent (peak) area. The so called "68-95-99.7 empirical rule" was adopted for selecting the frequent cycles. This rule assumes that 68.27%, 95.45% and 99.73% of the data, cycles, fall within 1, 2 and 3 standard deviations respectively from the mean (μ_d). This is illustrated in Figure 8.5. The mean (μ_d) and standard deviation (σ_d) values was calculated using *fitdist*[1], a built-in function in the MATLAB software package.

[1]https://uk.mathworks.com/help/stats/fitdist.html



(a) Class# 1 (b) Class# 2

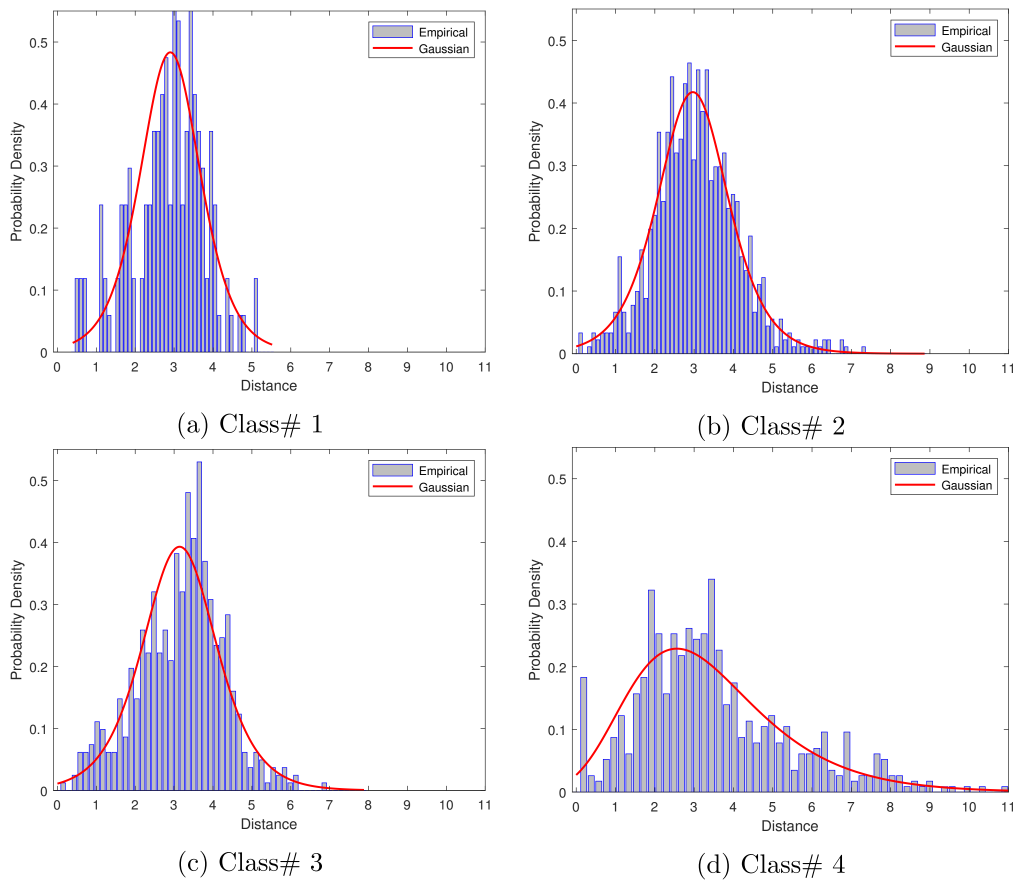
(c) Class# 3 (d) Class# 4

Figure 8.3: Example motifs (cycles) for each class in the Amplitude-PCG evaluation data generated using the proposed CE-FCS approach

Experiments were conducted using $\zeta = \{1, 2, 3\}$ to identify the most appropriate parameter settings (more details are given in Sub-section 8.5.4). Using the three values for ζ, the data set size, the set of cycles H, was reduced by approximately 32%, 5% and 1% respectively. The total percentage reductions were 45%, 18% and 13% of the original data size for $\zeta = \{1, 2, 3\}$ respectively. Given that the raw Amplitude-PCG time series comprised approximately 50 million data points, the remainder after application of the first pruning technique was 44.5 million data points. A further pruning was achieved with the second pruning technique; 24.5, 36.5 or 38.7 million data points respectively. The significance of the total reduction in the size of the input data was that it reduced the total number of motifs (sub-sequences, cycles) to be considered, which was not the case with respect to the previous three approaches presented in this thesis in Chapters 5 to 7.

8.5.3 Comparison with the SGR-FMD Approach

To evaluate the performance of the proposed CE-FCS approach presented in this chapter, its operation was compared with the SGR-FMD approach presented in Chapter 7, the



(a) Class# 1

(b) Class# 2

(c) Class# 3

(d) Class# 4

Figure 8.4: The best-fit distribution curve for distance similarity values for each of the four classes that appear in the evaluation data set

best approach presented in this thesis so far in terms of computational efficiency and effectiveness. To determine the runtime complexity, 18 sets of experiments were conducted using $\zeta = \{1, 2, 3\}$, $k = \{10, 20, 30\}$ and $flag = \{\text{all}, \text{max}\}$; $max = 2 \times k$ was again used as in the case of the SGR-FMD approach. Later experiments, detailed in Sub-section 8.5.4, demonstrated that the adopted range of values for both λ and σ_m had no effect on runtime; although for the experiments reported in this sub-section $\lambda = 17e5$ and $\sigma_m = 0.1$ were used. Therefore, the discussion of the runtime experiments presented here focuses on the ζ, k and $flag$ parameters (the pruning option used).

The recorded CE-FCS runtime results are presented in terms of milliseconds in Table 8.2, these are the average runtime for one PCG time series obtained using Five-fold Cross-Validation (FCV). In the table, the runtime results for the CE-FCS approach are presented for the Cycle Extraction pre-process (PCG_{ce}) and the Frequent Cycle Selection process (FCS) comprised of the Candidate Frequent Cycle Detection and Frequent Motif Discovery sub-processes. The second sub-process has two versions, "all" where all frequent cycles are considered, and "max" where only a predefined maximum number of frequent cycles are considered. The last column in the table (Total) presents the total runtime to process a single PCG time series (the sum of the values in the previous three columns).

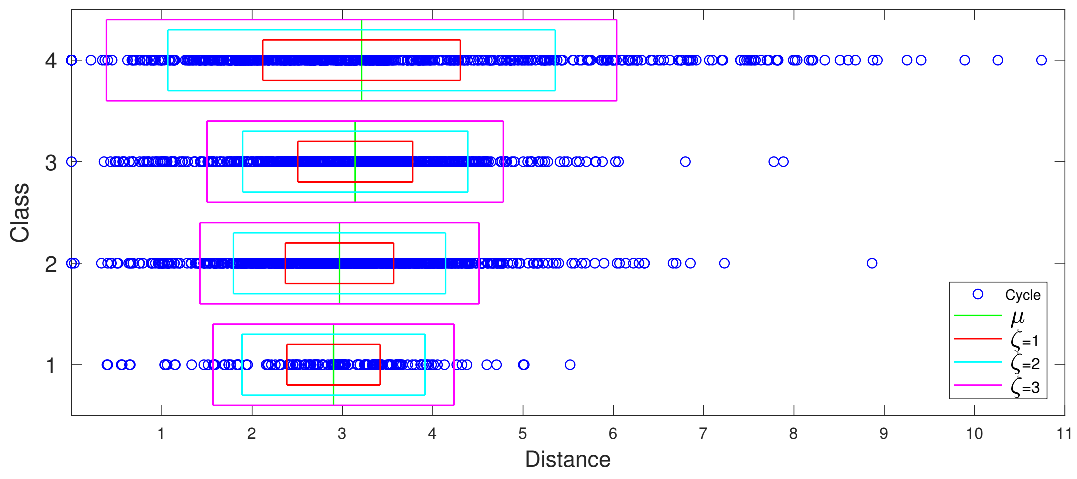


Figure 8.5: Distance distribution of the four classes that appear in the evaluation data set showing the first three standard deviations

Table 8.2: Detailed runtime results for CE-FCS approach (in milliseconds)

ζ	k	PCG$_{\text{ce}}$	FCS			Total	
			CFCD	FMD			
				all	max	all	max
1	10	270.68	69.15	178.81	255.93	518.64	595.76
	20			197.80	193.22	537.63	533.05
	30			210.68	188.47	550.51	528.30
2	10		69.49	205.08	507.46	545.25	847.63
	20			236.27	416.61	576.44	756.78
	30			259.15	403.05	599.32	743.22
3	10		69.66	201.19	606.27	541.53	946.61
	20			237.63	531.19	577.97	871.53
	30			265.59	514.24	605.93	854.58

Key: CFCD = candidate frequent cycle detection and FMD = frequent motif discovery.

From the table, it can be clearly seen that the difference between the runtime results using different values of ζ, when selecting candidate frequent cycles, was negligible. Moreover, the larger the k value, the more time that was required to discover the frequent motifs using the "all" pruning option, and the less time that was required to discover the frequent motifs using "max" pruning option. The difference between the pruning options ("all" and "max") in runtime was not very clear: most "max" experiments required longer runtime; however, two of them ($\zeta = 1$ with $k = 20$, and $\zeta = 1$ with $k = 30$), achieved lower runtimes than the "all" option. It is more obvious when using the "max" option that the required runtime increased with ζ because a larger number of included cycles requires more processing. However, the total runtime required for a single time series to be processed, on average, was similar in all cases; the difference in runtime was less than half a second, the fastest was 518.64 ms (0.52 sec) and the slowest was 946.61 ms (0.95 sec). Comparing these runtime results with the SGR-FMD approach, the runtimes required for both approaches are presented in Table 8.3, these are the total runtimes to classify one record in s.SS format (seconds and decimals of seconds). From the table, it can clearly be seen that the runtime for the CE-FCS approach was much better than that required using the SGR-FMD approach. It took, on average, less than a second per record using CE-FCS, this compared very favourably with the SGR-FMD approach, which required, on average, more than two and a half seconds per record. The proposed CE-FCS approach therefore reduced the model generation time by a factor of more than three.

Table 8.3: Recorded runtime results (s.SS format) for the CE-FCS and SGR-FMD approaches

ζ	k	CE-FCS Approach		ω	k	SGR-FMD Approach
		all	max			
1	10	0.52	0.60	100	10	0.49
	20	0.54	0.53		20	0.81
	30	0.55	0.53		30	1.13
2	10	0.55	0.85	200	10	1.17
	20	0.58	0.76		20	2.19
	30	0.60	0.74		30	3.22
3	10	0.54	0.95	300	10	2.33
	20	0.58	0.87		20	4.52
	30	0.60	0.85		30	6.71
Average		0.65		2.51		

To compare with the SGR-FMD approach in terms of effectiveness, the classification metrics for both approaches are presented in Table 8.4. In the table, the first column indicates the adopted NNC Conflict Resolution Method (CRM), Shortest Distance (SD), Shortest Total Distances (STD) and Highest Votes (HV). Details of these CRMs were given in the previous chapter, Chapter 7, where the SGR-FMD approach was presented. For the proposed approach, only the experiments that used $\lambda = 164e5$, $\sigma_m = 0.1$, $k_{nnc} = 3$ and "max" pruning option are presented in the table, because this combination produced the best classification accuracy; for the SGR-FMD approach, only the experiments that used $k_{nnc} = 1$ are presented for the same reason. The best recorded accuracy for the CE-FCS approach was 72.0%, whereas 71.3% was the best recorded accuracy with respect to the SGR-FMD approach. The precision, recall and F-score values were 0.253, 0.311 and 0.272 for the CE-FCS approach, and 0.458, 0.442 and 0.440 for the SGR-FMD approach. Using the best accuracy results, the CE-FCS approach required 0.74 second to mine one record, whereas the SGR-FMD approach required 0.49 second. However, most of the runtime for the CE-FCS approach was for the FCS processes which would only be conducted in the training stage, whilst in the application stage, only PCG_{ce} is needed, which required 0.27 seconds on average. The increase in runtime, using CE-FCS, for the training stage was a matter of milliseconds (a quarter of a second) for an increase in accuracy of 72.0% versus 71.3%, and a decrease in the application runtime of 0.27 sec versus 0.49 sec.

Table 8.4: Recorded accuracy, precision, recall and F-score for the CE-FCS and SGR-FMD approaches

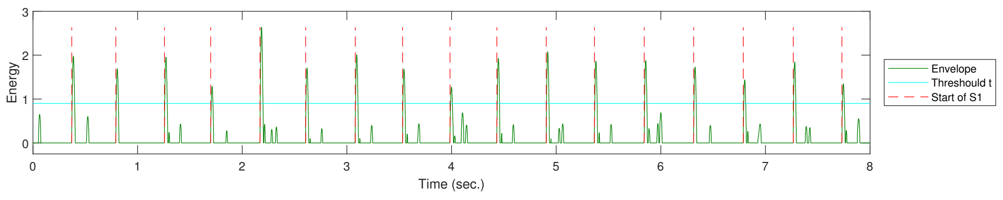
CRM	CE-FCS Approach						SGR-FMD Approach					
	ζ	k	Acc	Prec	Rec	F-s	ω	k	Acc	Prec	Rec	F-s
		10	0.618	0.176	0.263	0.189		10	0.693	0.310	0.356	0.329
	1	20	0.592	0.145	0.211	0.151	100	20	0.686	0.192	0.248	0.213
		30	0.592	0.145	0.211	0.151		30	0.711	0.208	0.271	0.215
		10	0.601	0.053	0.250	0.086		10	0.643	0.278	0.319	0.297
SD	2	20	0.627	0.229	0.265	0.231	200	20	0.636	0.355	0.231	0.274
		30	0.626	0.233	0.265	0.234		30	0.643	0.274	0.234	0.245
		10	0.609	0.149	0.216	0.156		10	0.654	0.211	0.275	0.225
	3	20	0.575	0.098	0.146	0.111	300	20	0.636	0.150	0.227	0.179
		30	0.602	0.147	0.191	0.157		30	0.657	0.339	0.347	0.329
		10	0.609	0.151	0.221	0.173		10	0.660	0.347	0.339	0.338
	1	20	0.575	0.157	0.178	0.140	100	20	0.602	0.190	0.218	0.202
		30	0.594	0.216	0.228	0.207		30	0.561	0.112	0.166	0.124
		10	0.618	0.208	0.254	0.221		10	0.635	0.213	0.289	0.244
STD	2	20	0.585	0.115	0.201	0.129	200	20	0.642	0.262	0.301	0.268
		30	0.602	0.153	0.234	0.167		30	0.575	0.165	0.124	0.134
		10	0.601	0.172	0.246	0.192		10	0.637	0.158	0.250	0.184
	3	20	0.567	0.159	0.176	0.166	300	20	0.615	0.148	0.198	0.168
		30	0.575	0.201	0.144	0.147		30	0.613	0.240	0.213	0.212
		10	0.617	0.076	0.270	0.114		10	**0.713**	0.458	0.442	0.440
	1	20	0.626	0.083	0.279	0.123	100	20	0.686	0.150	0.226	0.172
		30	0.626	0.087	0.273	0.125		30	0.711	0.105	0.250	0.147
		10	0.650	0.180	0.283	0.211		10	0.651	0.192	0.231	0.206
HV	2	20	0.712	0.241	0.301	0.261	200	20	0.645	0.197	0.253	0.220
		30	**0.720**	0.253	0.311	0.272		30	0.670	0.277	0.263	0.265
		10	0.660	0.132	0.248	0.171		10	0.656	0.162	0.275	0.193
	3	20	0.679	0.172	0.275	0.200	300	20	0.636	0.158	0.236	0.185
		30	0.687	0.147	0.255	0.176		30	0.710	0.296	0.321	0.300

8.5.4 Most Appropriate Parameter Settings

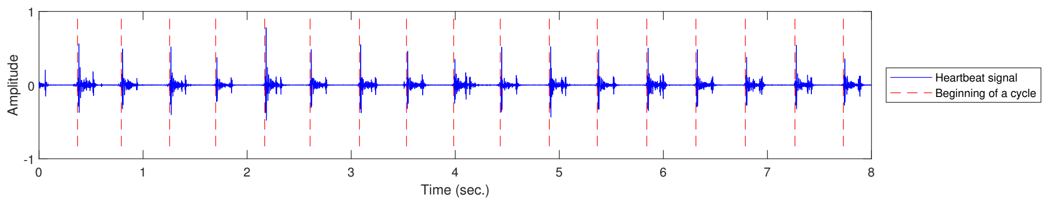
As shown in the previous section, Section 8.4, the proposed approach used a number of user-defined parameters for: (i) the PCG_{ce} pre-process, (ii) the FCS process and (iii) the classification model. A set of values for each parameter was experimented with to determine the best value. These are discussed, with respect to the above listed processes, in the following three sub-sections, Sub-sections 8.5.4.1 to 8.5.4.3 respectively.

8.5.4.1 Parameter Settings for the PCG_{ce} Technique

The proposed cycle extraction technique uses a dynamic cut-off value t, computed using a user-specified $\alpha\%$ threshold expressed as a percentage of the width of the oscillation with the highest Amplitude in a given PCG time series. This method was also adopted in [58], where $\alpha = 70$ was suggested to detect the S1 and S2 PCG components, and $\alpha = 90$ to detect possible“click”. The focus with respect to this chapter was the detection of the start of S1, therefore $\alpha = 70$ was used which was able to detect all S1 components (and some S2 components which were discarded later). An example of the results obtained is given in Figure 8.6 using a fragment of one of the PCG time series used for the evaluation. In Figure 8.6(a), the envelope signal is given for the raw signal shown in Figure 8.6(b). From the figure, it can be seen that all S1s are identified (and in this case no S2s) but no noise points. Using $\alpha = 70$, applied to all PCG recordings in the data set, resulted in the extraction of 2,139 cardiac cycles; an average of 36.25 cycles per PCG time series.



(a) Envelope PCG signal with “cut-off” line



(b) Raw PCG signal with “start points” marked

Figure 8.6: Extraction of cardiac cycles from the envelope of signal energy using the PCG_{ce} technique

8.5.4.2 Parameter Settings for the FCS Process

Recall that the proposed Frequent Cycle Selection process, FCS, required six parameters: (i) the ζ a threshold to determine the number of standard deviations to be used, (ii) max to define the maximum number of candidate frequent motifs to be considered, (iii) k to define the maximum number of frequent motifs to be selected ($k < max$), (iv) the λ similarity threshold, expressed in terms of the maximum distance between two motifs, (v) the σ_m frequency threshold, expressed in terms of the minimum percentage of possible motifs and (vi) a $flag$ to determine the pruning option (“max” or “all”).

The selected values for these parameters all affected the number of frequent motifs identified and consequently the quality of any further utilisation of the motifs. Clearly, the higher the σ_m value, the fewer the number of motifs that would be identified, because the criteria for frequency would become stricter as σ_m is increased. On the contrary, the higher the λ value, the greater the number of motifs that would be identified because the criteria for similarity would become less strict as λ is increased. Moreover, as the value for ζ is increased, the number of selected motifs would increase but the average frequency of occurrence would decrease (as illustrated previously in Figure 8.4). The values for max

and k would also affect the number of identified candidate frequent motifs (cycles) and, it was conjectured, would thus also influence the number of frequent motifs eventually selected.

To identify the most appropriate parameter settings, a range of three values for both ζ and k were considered, $\{1, 2, 3\}$ and $\{10, 20, 30\}$ respectively. The value for max was again set to $2 \times k$ although any value greater than k could have been used. The set of ζ values are axiomatic values in the Gaussian distribution, while the k and max values were chosen according to a set of experiments, not reported here, which showed that when $max > 60$, the number of motifs in some classes did not exceed 60, therefore this value was chosen to be the highest value in the selected set of values. With regard to the pruning option, $flag$, early experiments indicated that when using the "max" option some classes had no motifs associated with them. A modification was therefore made to the "max" procedure to ensure that each class had at least one motif associated with it. This change solved the problem and it is the reported algorithm above in Section 8.4.

A range of twenty values for λ was also experimented with, from d_{min}, which is the minimum distance between all motifs in different classes, to d_{max}, which is the maximum distance between all motifs in different classes, increasing in steps of $(d_{max} - d_{min})/19$. A range of eight values for σ_m was also considered, these values were derived using the formula: $\sigma_m = p \times 10^q$, where $p = \{1, 5\}$ and $q = \{-2, -1, 0, 1\}$; this formula was used for producing a variety of values in ascending order from 0.01% and as big as 50.0%. Both λ and σ_m were considered to be more significant with respect to the performance of the proposed approach, and thus a greater number of values was considered compared to the range of values considered for ζ, k and max.

For the experiments to determine the most appropriate parameter settings for λ and σ_m, both pruning options, "all" and "max", were considered. The results are presented in graph form in Figure 8.7. The first three columns in the grid represent a range of values for the max parameter, $\{20, 40, 60\}$, applicable when using the "max" option; the final column shows the results obtained when using the "all" option. The grid rows indicate the calculated σ_m settings, in ascending order. For each graph, the Y-axis represents the Number of Retained Motifs (NRM), the number of frequent motifs that were retained on completion of the max selection process. The X-axis indicates the λ value. Inspection of Figure 8.7 indicates that the results can be grouped into three clusters according to σ_m value:

- Low ($\sigma_m = \{0.01, 0.05, 0.1\}$) Rows 1, 2 and 3 in the grid given in Figure 8.7.
- Medium ($\sigma_m = \{0.5, 1\}$) Rows 4 and 5 in the grid.
- High ($\sigma_m = \{5, 10, 50\}$) Rows 6, 7 and 8 in the grid.

The behaviours associated with these three σ_m defined clusters are summarised in Figure 8.8. The figure shows three NRM plots, one for each of the three clusters: low

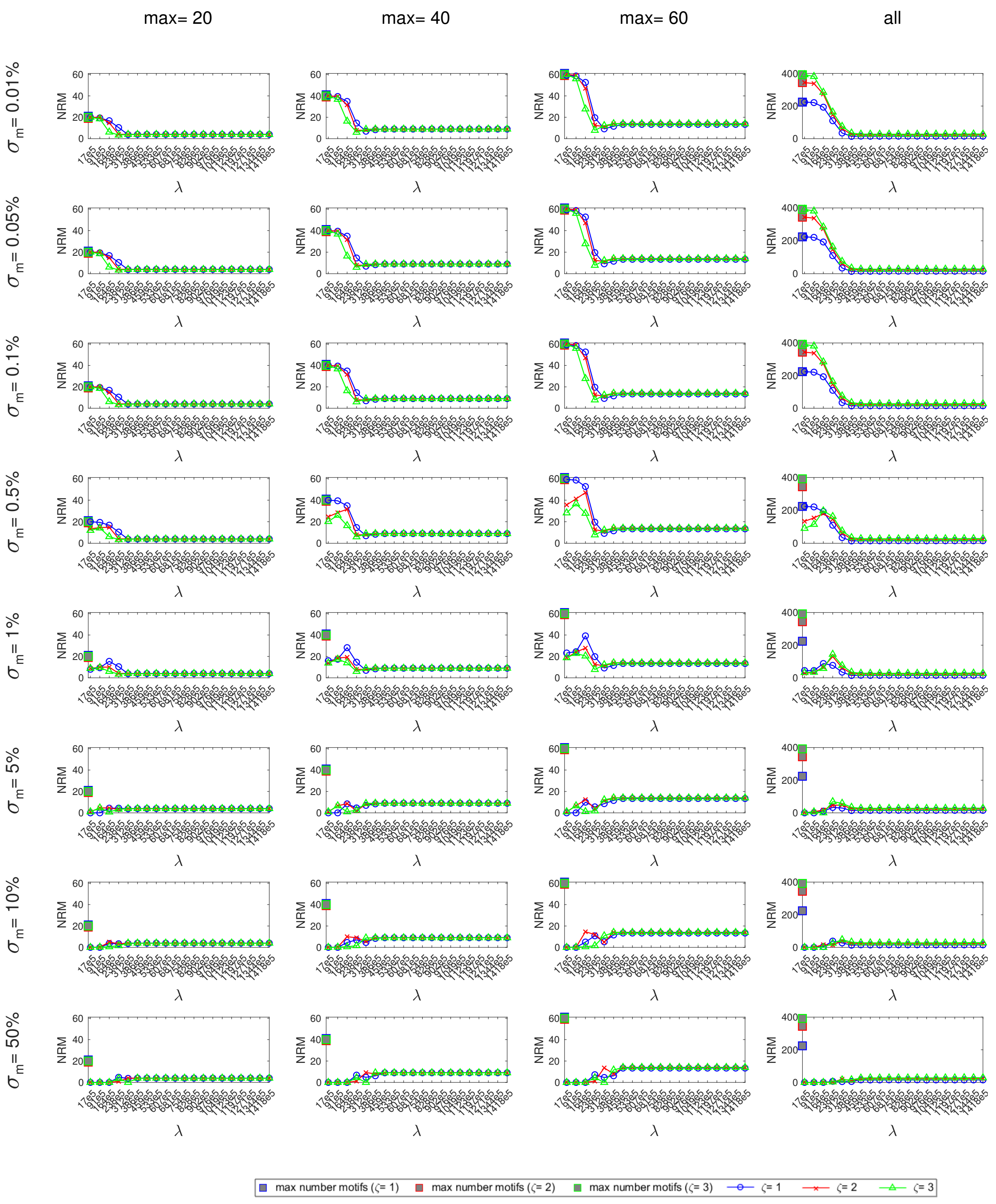


Figure 8.7: Average Number of Retained Motifs (NRM) using different values for the thresholds λ, σ_m, ζ, $flag$ and max. The twenty λ values start with $17e5$ and end with $1418e5$, for a clear view of the X-axis, see Figure 8.8

(blue), medium (green), high (magenta). The X-axis represents the 20 λ values, where the first value represents d_{min} and the last value represents d_{max}. The NRM values are referenced by the Y-axis, each cluster plot is given with reference to the maximum number of motifs indicated by a grey box on the Y-axis. In the figure, the top five best σ_m-λ combinations are highlighted with red circles. Considering the NRM plot lines first, from the figure, for the "high" cluster, it can be seen that the NRM value increases as λ increases, reaching a peak followed by a small drop, before stabilising. The "medium" cluster displays a similar behaviour. The "low" cluster displayed a different behaviour, the highest recorded NRM value was recorded with a low λ value after which the NRM values dropped off. It is conjectured that there is a behaviour somewhere between the centroids of the "medium" and "high" clusters where the NRM plot changes from increasing at the start to decreasing.

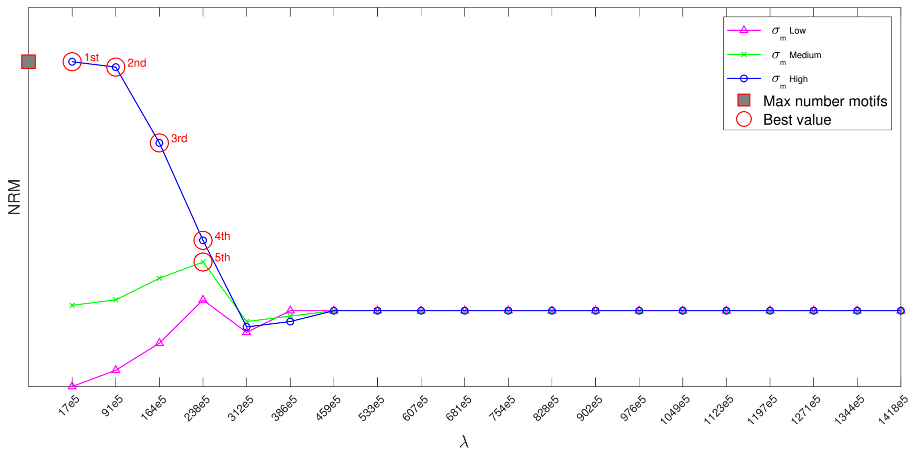


Figure 8.8: Summary of graphs presented in Figure 8.7 with graphs grouped according to associated σ_m values (low, medium or high)

Returning to the "Best value" points in Figure 8.8, these are the σ_m-cluster, λ combinations associated with the top five best NRM values. The set of combinations was: $\{\langle low, 17e5\rangle, \langle low, 91e5\rangle, \langle low, 164e5\rangle, \langle low, 238e5\rangle, \langle medium, 238e5\rangle\}$. These were therefore the values used with respect to the further classification experiments reported on with respect to this chapter.

To summarise, for the further experiments reported on in the remainder of this chapter, the parameter settings used with respect to the FCS process were as follows:

- $\zeta = \{1, 2, 3\}$.
- $k = \{10, 20, 30\}$.
- $max = 2 \times k$.
- $\lambda = \{17e5,\ 91e5,\ 164e5,\ 238e5\}$.
- $\sigma_m = \{0.1, 1\}$.

- $flag = \{\text{all, max}\}$.

Using these values there are 90 combinations, hence 90 sets of experiments were conducted. Given that there are five combinations of λ and σ_m: all four λ values were combined with $\sigma_m = 0.1$ and the last λ value with $\sigma_m = 1$ (for reasons established above); in other words, the five combinations of λ and σ_m are: $17e5, 0.1$; $91e5, 0.1$; $164e5, 0.1$; $238e5, 0.1$ and $238e5, 1$. The results obtained are given in Appendix C in Tables C.2 to C.11. The results using the best performing parameters, with regard to the λ and σ_m combination, are presented in Table 8.5. In the table, the relevant λ and σ_m values are given in Columns 1 and 2, and the Candidate Frequent Cycle Detection and Frequent Motif Discovery parameters in Columns 3, 4, 5, 6 and 7. The relevance of the Conflict Resolution Method (CRM), Column 4, will be discussed further in the following sub-section, Sub-section 8.5.4.3. The results, in terms of accuracy and runtime, are given in Columns 8 and 9. The runtime is the time to process an individual time series, and is given in seconds. The best recorded accuracy was 0.720 (72.0%) when $\lambda = 164e5$ and $\sigma_m = 0.1$; this was produced using the "max" pruning method, the HV conflict resolution method, $\zeta = 2$, $k = 30$ and $k_{nnc} = 3$, and took only 0.74 seconds per PCG time series.

Table 8.5: The best classification accuracies for the CE-FCS approach using different parameters

Thresholds		Parameters					Results	
λ	σ_m	$flag$	CRM	ζ	k	k_{nnc}	Acc	Runtime
17e5	0.1	all	HV	3	10	1	0.667	0.54
91e5	0.1	max	HV	3	20	3	0.704	0.87
164e5	0.1	max	HV	2	30	3	0.720	0.74
238e5	0.1	max	HV	2	30	3	0.695	0.74
238e5	1	max	HV	2	30	3	0.695	0.74

8.5.4.3 Parameter Settings for the Classification Model

As mentioned in the objectives for the evaluation, the adopted process for classifying previously unseen cycles (motifs) was NNC. As in all previous approaches proposed in this thesis, the data set was divided into a training set T and a testing set using FCV. The accuracy of the classification would then be an indicator of the quality of the proposed CE-FCS approach; the metrics used were accuracy (acc), precision (prec), recall (rec) and F-score (f-s).

The NNC model, considered with respect to the evaluation presented here, classifies an unseen cycle according to the k_{nnc} most similar existing (labelled) motifs held in the "motif bank". Similarity was measured using Euclidean Distance (ED) measurement. More specifically, with respect to the evaluation presented here, the bank comprised a set of motifs, $H'' = \{\langle m_1, c_1\rangle, \langle m_2, c_2\rangle, \dots\}$ where m_i is a frequent motif (cycle) and c_i is a class-label. A new query, PCG time series, is processed using the proposed PCG_{ce}

pre-processing algorithm (see Section 8.3), so that it is represented as a set of cycles, $H = \{h_1, h_2, \dots\}$ each of which is to be matched with the motifs held in H''. In other words, h_i will be labelled based on the k_{nnc} most similar m_i in H''. Two values for the number of nearest neighbours to be identified were used, $k_{nnc} = 1$ and $k_{nnc} = 3$.

Given that each query PCG to be labelled usually comprises a number of cycles, each of which will be labelled separately, there is a chance that more than one class-label will be associated with the query PCG. However, only a single, most appropriate class-label is required. To select the "winning" class-label, three Conflict Resolution Methods (CRMs) were proposed with respect to the SGR-FMD approach described in the previous chapter, Chapter 7: (i) Shortest Distance (SD), (ii) Shortest Total Distances (STD) and (iii) Highest Votes (HV). These CRMs were also experimented with in the context of the CE-FCS approach presented in this chapter.

From the foregoing, experiments were conducted using two values for k_{nnc}, $\{1, 3\}$, and three CRMs, {SD, STD, HV}. Performance was recorded in terms of accuracy, precision, recall and F-score. For the experiments, $\zeta = \{1, 2, 3\}$ and $k = \{10, 20, 30\}$ were again used, coupled with the five combinations of λ and σ_m: $17e5, 0.1$; $91e5, 0.1$; $164e5, 0.1$; $238e5, 0.1$ and $238e5, 1$; and $max = 2 \times k$. This resulted in 90 combinations, hence 90 sets of experiments, each of which could be coupled with the two considered values for k_{nnc} and three CRMs; thus 540 sets of results (acc, prec, rec and f-s) were produced.

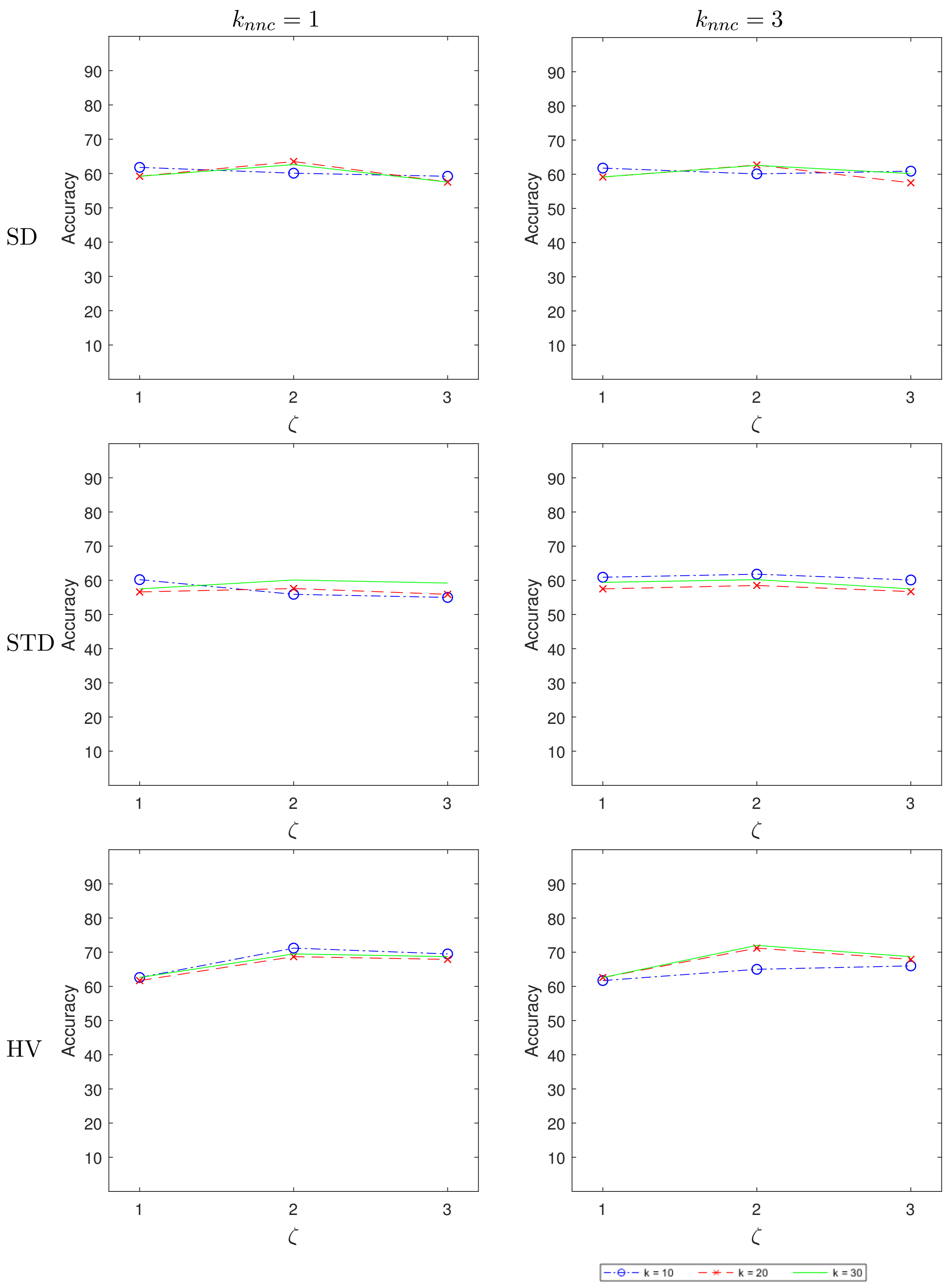
The results obtained are given in 10 tables in Appendix C, Tables C.2 to C.11. Among these results, there are some interesting patterns. For $\lambda = 17e5, 91e5$ and $\lambda = 164e5$, HV always produced the worst accuracy, with $\zeta = 1$, regardless of the k_{nnc} value used. The other two CRMs, SD and STD, almost always produced the worst accuracy when $\zeta = 3$ and $\lambda = 238e5$. However, $\lambda = 238e5$ and $\zeta = 2$ always produced the best accuracy regardless of the k_{nnc} value, $flag$ and CRM used. Furthermore, it was found that when using the values $\{0.1, 1\}$ for σ_m, there was no difference in terms of accuracy, runtime or the other metrics considered (prec, rec and f-s). From a broader perspective, the 10 best accuracy results obtained from the 10 tables were obtained using the HV conflict resolution method, most using $k_{nnc} = 3$.

To better compare the classification model parameter settings, the accuracy results of the best performing combination, $\lambda = 164e5$, $\sigma_m = 0.1$ and the "max" pruning method; are presented in graph form in Table 8.6. Inspection of the results indicates that accuracy was around 60% for SD and STD, and increased to peak above 70% for HV regardless of the k_{nnc} value used. Given that HV also produced the 10 best accuracy results for the 10 combinations of λ, σ_m and CRM, it can be concluded that the HV conflict resolution method was the most appropriate.

8.6 Discussion

Given the results presented in the previous section, it can be concluded that, when using the CE-FCS approach, the PCG data volume was reduced from 55% to 87% of its orig-

Table 8.6: Classification accuracy for the CE-FCS approach using three conflict resolution methods, two values for k_{nnc} and the parameter settings that produced the best accuracy



inal volume, depending on the number of standard deviations used (the ζ value). The adopted motif generation process, FCS, offered a number of significant advantages leading to a dramatic reduction in the total number of motifs compared with the previous three approaches considered in this thesis, in Chapters 5 to 7. All of these improvements were reflected in the recorded runtime results, which were reduced by a factor of more than three. This runtime advantage was not the only benefit offered by the proposed CE-FCS approach, another was the increase in accuracy, which reached 72%, the best recorded accuracy in this thesis. Using the CE-FCS approach to extract the k most frequent motifs from a PCG time series record required less than a second on average; whilst in the case of the SGR-FMD approach, an average of two and a half seconds was required to extract the k most frequent motifs from a PCG time series record.

Given the effect of the similarity and frequency thresholds on accuracy results, it can be observed that accuracy might be increased if an alternative similarity function were used. Given that the motifs in the CE-FCS approach are cycles, whose envelopes were calculated using the Cycle Extraction process (PCG_{ce}), these envelopes could be isolated and then the FCS process applied with Dynamic Time Warping (DTW) similarity measurement instead of Euclidean Distance similarity measurement. DTW offers the advantage that two cycles can be “warped” so that a best similarity distance can be found. It was anticipated that this would improve the similarity measurement, and hence the overall classification performance. However, this idea is not explored further in this thesis and instead is left as an item for future work.

The best classification accuracy results obtained by all four approaches considered in this thesis, (i) the MK Benchmark, (ii) PCG_{seg} Classification, (iii) SGR-FMD and (iv) CE-FCS, are presented in Table 8.7. For each approach, the best accuracy result, the runtime required to obtain the accuracy, divided between the training stage and the application stage, and the associate F-score are given. Inspection of the table indicates that the CE-FCS approach produces the overall best results. CE-FCS features the same accuracy as the PCG_{seg} Classification approach, but achieved in a matter of milliseconds; a significant decrease in the application time. For the application domain considered in this thesis, diagnosing mitral valve disease using an electronic stethoscope, a processing time of less than one second, with 72% accuracy, is an encouraging result.

Table 8.7: The best classification performance for all four approaches considered in this thesis

Approach	Acc	Runtime		F-s
		Training	Application	
Benchmark	0.711	51:11.02	51:11.02	0.270
PCG_{seg} Classification	0.720	07:00.39	07:00.39	0.211
SGR-FMD	0.713	00:00.49	00:00.49	0.440
CE-FCS	0.720	00:00.74	00:00.27	0.272

8.7 Summary

This chapter has presented the CE-FCS approach for PCG time series classification. The objective of the proposed approach was to address the challenge of finding frequent discriminative motifs in long time series in a manner that surpassed the previous approaches considered in this thesis. Two mechanisms were incorporated into the proposed CE-FCS approach that served to enhance performance. This comprised two processes: (i) Cycle Extraction and (ii) Candidate Frequent Cycle Detection. The first was derived from similar work, reported in the literature in the context of other application domains, and used to support motif generation. The second featured a novel way of using the statistical distribution of candidate frequent motifs, without comparing all cycles with all other cycles, to identify the most frequently occurring cycles; the Candidate Frequent Cycle Detection sub-process. The retained cycles were then further processed to extract motifs, the Frequent Motif Discovery sub-process. The performance of the proposed CE-FCS approach was ascertained in the context of runtime and the quality of the motifs identified; runtime was improved by a factor of more than three. The evaluation indicated that the CE-FCS approach was the most appropriate of the four approaches considered in this thesis. Further discussion of this observation is given in the following chapter, where the thesis is concluded with a summary and an overview of the main findings in terms of the original research question and subsidiary research questions postulated in Chapter 1. The following chapter also considers some potential directions for future work whereby the work presented in this thesis can be extended.

Chapter 9

Conclusions and Future Work

9.1 Introduction

This chapter concludes the work described throughout this thesis. The chapter commences, Section 9.2, with a summary of the material presented so far. This is followed, in Section 9.3, by the main findings and contributions of the work presented with respect to this thesis in the context of the overriding research question, and its subsidiary research questions, presented in Chapter 1. The chapter is concluded, in Section 9.4, with some suggestions regarding further potential areas for future work that might build-on the work provided in the thesis.

9.2 Summary of Thesis

The work presented in this thesis commenced with a scene-setting chapter, Chapter 1, where the "thesis map" was provided, including the motivations, the research question and its subsidiary questions, the main contributions of the thesis and the adopted research methodology. The theme of the thesis was Phonocardiogram classification for clinical diagnosis. The motivation for Phonocardiogram classification was the high percentage of mortality and morbidity worldwide because of heart diseases [47, 163]. It is also one of the most expensive medical conditions to treat. In addition, its diagnosis requires specialists and expensive special equipment not provided in primary care clinics. The overwhelming majority of cases diagnosed by primary care physicians for referral to cardiologists are for healthy subjects [164]. One way of minimising faulty diagnosis and therefore the total cost, not to mention other negative effects, such as disease worsening and weak productivity (in the case of false negatives), and frustration and fear in patients (in the case of false positives), is to provide Artificial Intelligence support for the diagnosis process. The need for AI technology in healthcare was identified in the UK Life Sciences Industrial Strategy report [24]. Electronic stethoscopes, a recent innovation, are able to provide a record of heartbeat activity (PCG). The fundamental idea presented in this thesis is that PCG signals can be fruitfully exploited, by applying time series analysis techniques to PCG data, to address the issues associated with heart disease identified above. PCG time series are typically very large, given the standard computing power available in doctors' surgeries, this presents a computational challenge; one way of addressing this issue is by identifying *motifs* within the time series [36, 39, 50, 150, 165, 171, 176]. However, finding motifs that are good representatives of class-labels, which can then be used to label (classify) previously unseen time series, is also computationally challenging. Therefore, the work presented in this thesis investigated techniques where by PCG classification, using the concept of motifs, could be conducted using a limited computational resource. Ideally, we would like "point of care" diagnosis using machine learning software shipped with digital stethoscopes; what might then be referred to as *intelligent digital stethoscopes.* This is the central motivation for the work presented in this thesis.

The thesis then went on, in Chapter 2, to give a review of the relevant previous work. The chapter commenced with an overview of time series analysis by considering: (i) time series supervised learning techniques (time series classification); (ii) the literature concerning PCG classification, the application domain of interest with respect to this thesis; and (iii) the idea of motif discovery, the particular time series analysis technique that dominates the work presented in this thesis, noting that the motif concept has not previously been applied to PCG data (to the best knowledge of the author). The chapter then reviewed common solutions presented in the literature to the challenge of time series analysis due to data volume; namely the pre-processing of the data with a particular focus on segmentation, because of its significance with respect to this thesis. The chapter was concluded with a review of the common evaluation methods used in the literature to measure classification model performance.

The following short chapter, Chapter 3, gave a description of the PCG data sets with respect to the evaluations reported on later in this thesis. The chapter started with a review of the application domain used as focus for the work presented in this thesis, building on discussion from earlier chapters. The chapter then went on to consider the existing PCG data sets that have been referenced in the literature and that are available for download. Next, a comprehensive overview was provided of the bespoke canine PCG data set specifically collected to support the work presented in this thesis.

Chapter 4 presented a formalism for Phonocardiogram classification using time series. Intuitively, a Phonocardiogram time series is a sequence of continuous real-valued data points representing Sample or Amplitude sound waves. It was noted that Phonocardiogram time series could be represented using the WAVE audio file format whereby each PCG point was conceptualised as a temporal event (p_i) where i is a time stamp and p is the Amplitude- or Sample-PCG value.

The following four chapters, Chapters 5 to 8, presented a sequence of time series-based Phonocardiogram classification approaches of increasing sophistication directed at Amplitude- and Sample-PCGs. The time series analysis technique used in these chapters was motif discovery; however, two different strategies for defining the motif were used, similarity and frequency. All these four chapters were structured in a similar manner comprising proposed approach, evaluation and discussion sections.

Chapter 5 introduced the first Phonocardiogram classification approach considered in this thesis, namely the MK Benchmark PCG Classification approach. The principle idea of the Benchmark approach was to provide a vehicle with which to compare the alternative approaches presented in this thesis. Any alternative approach, to be of merit, would need to be better than the benchmark. The idea underpinning the work presented in this chapter was to analyse the operation of the Benchmark approach in the context of the PCG application domain. The intention was then to develop a number of refined variations according to the outcome of the analysis. For this purpose, the Benchmark approach was directed at both the Amplitude- and Sample-PCG representations. The evaluation of the Benchmark approach was conducted using the two representations and a comparison was performed using a number of performance measurements. The evaluation indicated that the use of the Sample-PCG representation was more effective than Amplitude-PCG representation; thus the former was adopted with respect to the approach described in the following chapter. What was also clear from the evaluation was that some form of pre-processing of the data was an essential requirement if the runtime issue was to be addressed.

Chapter 6 presented the second Phonocardiogram classification approach, the PCG_{seg} Classification approach. Central to the approach was a novel bespoke segmentation technique, based on “shapes”, to reduce the overall size of the PCG time series, hence the runtime. The proposed segmentation technique was directed at the Sample-based PCG representation. This was because earlier work, described in Chapter 5 and as noted above, had demonstrated this to be the most appropriate representation (as opposed to the alternative Amplitude-based representation). The similarity function used with respect to the PCG_{seg} Classification approach was especially designed to operate with the proposed segmentation technique. The evaluation results of the proposed approach, compared to the Benchmark approach presented in the previous chapter, were good; runtime was reduced by a factor of more than eleven. The evaluation also suggested that by removing sub-sequences from individual PCG time series, sub-sequences that were unlikely to be representative of any class-label, further runtime gains and improvements in the quality of the identified motifs might be achieved.

The thesis then continued with Chapter 7 which presented the third Phonocardiogram classification approach, the SGR-FMD approach. The two main principles of this approach were: (i) to identify frequent motifs and (ii) to prune the time series in order to reduce the runtime and identify good motifs. Three different categories of time series sub-sequences suitable for pruning were identified: (i) sub-sequences that exist in every time series and hence are not representative of any particular class, (ii) sub-sequences that appear so infrequently that cannot be deemed to be relevant and (iii) sub-sequences that appear across two or more classes and thus cannot be usefully employed to discriminate between classes. The aim of this approach was to address the challenge of finding discriminative motifs in long time series by proposing two pruning mechanisms: (i) silent gap removal and

(ii) candidate frequent motif generation. The first technique was adopted because little useful information could be extracted from the "silent gap". The second technique featured a novel way of clustering sub-sequences, without comparing all sub-sequences with all other sub-sequences, to identify the most frequently occurring motifs, namely using the "zero-motif" concept. This approach used the Amplitude-based PCG representation because it was the most appropriate representation for tracking (identifying) the silent gaps. A comparison was performed, using a number of performance measurements, which indicated that runtime was improved by a factor of more than 278, this was more than satisfying. The reported evaluation revealed that the use of the Amplitude-PCG representation showed promise with respect to Phonocardiogram classification which could then be fruitfully employed for PCG time series analysis. For improvement in the quality of the selected motifs, it was suggested that more meaningful motifs could be generated by isolating recognisable patterns in the input time series.

In Chapter 8, the fourth Phonocardiogram classification approach, the CE-FCS approach, was presented. The idea presented in this chapter was to pre-process the PCG data so as to generate meaningful motifs and, at the same time, reduce the computations needed later in the process of discovering frequent motifs. This was applied to the PCG signals by extracting the heart cycles that represented potential motifs using a technique referred to as the PCG_{ce} technique and by considering the statistical distribution of these motifs to select the most frequent among them. Because the reoccurring heart cycles in the PCG time series were the subject of interest, the used data representation for the evaluation in this chapter was again the Amplitude-PCG time series where the plots clearly feature the heartbeat cycles. The performance of the proposed approach was ascertained in the context of runtime and the quality of the motifs identified. This approach seems to be the best of the four approaches discovered in this thesis; its accuracy was the highest recorded and the application runtime was the shortest.

9.3 Main Findings and Contributions

This section revisits the postulated research question and the associated subsidiary research questions. They are discussed, in this section, in terms of the "main findings" of the work presented throughout this thesis. The motivation for this research was the need for AI support to provide point of care diagnosis by applying time series analysis techniques to PCG data. The fundamental idea promoted in this thesis was to use the concept of motifs to build a classification model. The challenge was how this can best be achieved; the research question to be answered was thus:

What are the most appropriate mechanisms that can be used to identify indicative patterns (motifs) in previously unseen PCG data in a manner that is both efficient and effective for the purpose of PCG data classification?

The resolution of this research question necessitated the resolution of a number of related subsidiary research questions. Therefore, prior to considering the main research question to be addressed, the responses to the subsidiary questions are considered first. Note that some of the responses are common to more than one subsidiary question because, in some cases, a range of aspects were taken into consideration. In other words, some of the solutions presented in this thesis served to provide answers to more than one subsidiary question. Also, some subsidiary questions had more than one answer. The four subsidiary research questions and the answers to these questions were as follows:

1. **Best "motif".**

   What is the most appropriate mechanism to identify "motifs" that are indicative of some conditions within the PCG time series of interest?

   The challenge of finding a mechanism that can be used to identify the most representative motifs of a class was addressed by considering a number of aspects of the subsidiary question as itemised below. Considering all these aspects, **the most appropriate technique to recognise representative motifs, discovered throughout this thesis, was to extract the heart cycles from the Amplitude-PCG time series using the $\mathbf{PCG_{ce}}$ technique and then to consider the statistical distribution of these cycles to select the most frequent among them using a technique referred to as the FCS process**. These two techniques were built into the CE-FCS approach presented in Chapter 8. This conclusion was reached after exploring a number of potential solutions:

   - Two strategies for determining whether a motif is representative of a class or not, one based on similarity and the other on frequency.
   - A novel way of representing the data points within a WAVE file, to form a series of "Samples" in place of "Amplitudes", the former was used for the first time with respect to the work described in this thesis. Both data representations, Amplitude- and Sample-based, were experimented with.
   - A tractable benchmark motif discovery algorithm, the MK approach.
   - A novel bespoke segmentation technique, PCG_{seg}, for Sample-based data.
   - Two novel similarity measurements, strict and tolerant for use with PCG_{seg}.
   - A novel "zero-motif" mechanism for removing sub-sequences that cannot be frequent, used with respect to the proposed SGR-FMD and CE-FCS approaches.
   - A "best of the best" approach where best performing motifs are identified and selected, and then the top k "best of the best" performing motifs are selected, as used in the SGR-FMD and CE-FCS approaches.
   - Two techniques to exclude parts of the input data: (i) silent gap removal (PCG_{sgr}) and (ii) avoidance of trivial matches (matches between sub-sequences that overlap), as used with respect to the four approaches.

- A two-level mechanism whereby globally discriminative motifs are identified by first considering motifs at the local level before moving on to the global level, as used in the SGR-FMD and CE-FCS approaches.
- A pre-processing technique to isolate cycles, meaningful reoccurring patterns in the data, as incorporated into the CE-FCS approach.
- A technique, built into the CE-FCS approach, that used all parts of the data to participate in motif discovery (the alternative techniques considered did not do this).

2. **Tractable process.**

   With regard to the answer to the first subsidiary question, how can this mechanism be applied in a tractable manner, given the challenge of processing large PCG data time series?

   The CE-FCS approach was made tractable by: reducing the overall size of the input data, reducing the number of computations and incorporating the idea of early abandonment. This was achieved using: (i) the "zero-motif" concept which enhanced the runtime by retaining only potential frequent sub-sequences and hence reducing the computations needed for comparing all sub-sequences with all other sub-sequences to identify the most frequently occurring motifs, (ii) the statistical distribution of the outcomes of the application of the zero-motif concept to identify frequent motifs and hence reduce the required runtime, (iii) motif selection using the "best of the best" idea which reduced the number of computations required, and (iv) the "early abandonment" concept whereby comparison computation is terminated as early as possible. These techniques were all incorporated into the CE-FCS approach to make it tractable. Using these techniques, the runtime for diagnosing one subject was less than a second.

3. **Best classification model.**

   How can the identified motifs be best used to generate a PCG classification model?

   This subsidiary research question was investigated using two classification models, the Nearest Neighbour Classification (NNC) model frequently used in the context of time series analysis, and the Smallest Average Classification (SAC) model. These models were experimented with using a range of parameter settings and three different Conflict Resolution Methods (CRMs): Shortest Distance (SD), Shortest Total Distances (STD) and Highest Votes (HV). **It was found that the most appropriate mechanism for generating a PCG classification model was the NNC model**. A best accuracy of 72% was obtained. Recall that the CRMs were used because any unlabelled time series to be classified, using the CE-FCS approach, would be represented by a number of motifs. Each one of these motifs had a class-label, which might be contradictory, when only a single class-label was required.

4. **Best usage.**

   How can the generated PCG classification model best be used to provide point of care, near real time, diagnosis?

   The challenge of finding a model that can be used to diagnose PCG data, without the support of accompanying ECG data, was addressed by conducting more than 1600 experiments using the four proposed approaches, (i) the MK Benchmark, (ii) PCG_{seg} Classification, (iii) SGR-FMD and (iv) CE-FCS approaches (Chapters 5 to 8). The CE-FCS approach (Chapter 8) was found to be the most accurate and provided the fastest in response time. Thus, **the CE-FCS approach was considered the best classification model among all the four approaches considered to address the issue of PCG signal diagnosis in real time at point of care**. For the application domain considered in this thesis, diagnosing mitral valve disease using an electronic stethoscope, the best response time achieved was in a matter of milliseconds coupled with an accuracy of 72%.

Returning to the overriding research question, the most appropriate mechanism for identifying indicative patterns in previously unseen PCG data was the CE-FCS approach presented in Chapter 8, the fourth main contribution of this thesis.

For completeness, the main contributions for the work presented in this thesis are restated from Chapter 1 as follows:

- **Contribution 1, The MK Benchmark PCG Classification Approach:** The principle idea of the Benchmark approach was to provide a vehicle with which to compare the three alternative approaches proposed later in the thesis. The aim was to analyse the operation of this approach in the context of the Phonocardiogram application domain. A number of techniques were used in this approach:

  1. A tractable, not approximate, algorithm (the MK algorithm [112]) for Phonocardiogram classification.
  2. The "Sample" representation method. The typical way of representing PCG signals is as Amplitude series; a novel method is to represent it as Sample series. The novel method showed a reduction in the data size by approximately half.
  3. The similarity-based representation strategy for identifying motifs that were "representative" of a class.
  4. An exclusion technique to exclude "trivial" comparisons from consideration during the motif discovery process as an in-process technique.

5. Process termination using the concept of “early abandonment” when conducting similarity comparison for classification purposes, hence providing efficiency gains.
6. A new classification model, Smallest Average Classification (SAC), developed by the author; a variation of Nearest Neighbour Classifier. It takes into consideration the similarity between a new motif to be labelled and all motifs for each class in the “motif bank”.

The evaluation of this approach indicated that some form of pre-processing of the data was an essential requirement if the runtime issue was to be addressed. This is discussed further in Chapter 5.

- **Contribution 2, The PCG$_{\mathbf{seg}}$ Classification Approach:** This approach was designed to address the issue of the overall size of the PCG time series to be processed. The techniques incorporated into this approach, to improve upon the benchmark approach in terms of efficiency and effectiveness, were as follows:

1. The “Sample” representation method, which reduced the overall data size.
2. A novel bespoke, two layer, time series segmentation technique, the PCG$_{\mathrm{seg}}$ technique, to transform the data representation. It was proposed to further reduce the data size; the number of segments was much less than the number of points.
3. Two novel mechanisms to measure similarity when using the proposed two-level hierarchical segmentation: Strict and Tolerant Similarity Measurement.
4. A process termination technique for use with the proposed two-level segmentation to allow early abandonment to avoid the need of going down to the bottom level of the segmentation if the top level did not satisfy prescribed similarity conditions.
5. The similarity-based representation strategy, the exclusion technique for “trivial” matches, “early abandonment” and the SAC classification model as proposed with respect to the Benchmark approach.

The PCG$_{\mathrm{seg}}$ approach improved the runtime (by a factor of more than eleven) without adversely affecting the resulting accuracy. This is discussed further in Chapter 6.

- **Contribution 3, The SGR-FMD Approach:** This approach was designed to reduce the complexity of the motif discovery process by pre-processing the data in preparation for motif discovery. The fundamental idea was to prune the time series by removing sub-sequences that were unlikely to be representative of any class-label. This approach used the following techniques to enhance the quality criteria:

1. A frequency-based representation strategy for identifying “representative” motifs of a class.

2. In addition to the exclusion technique for "trivial" matches, an exclusion technique, to enhance motif selection, used as a pre-processing technique, the PCG_{sgr} technique. It excluded some parts of the data on the grounds that they were irrelevant to the analysis task at hand; namely, excluding the silent gaps that appear in the PCG signals.
3. Two levels of representativeness were considered: local and global. The first was widely used in the literature, where a motif is selected at the single time series level. Using the local technique, time series that might be the result of background noise could be selected as being representative of a class. Given an unseen record to be classified which features the same background noise, the record could be labelled based on that noise. The local technique therefore has some disadvantages. Consequently, by using global representativeness the effect of unwanted noise could be avoided.
4. A pruning technique used the novel "zero-motif" concept; a base motif with which all sub-sequences in the data can be compared. This technique retained the potential discriminative sub-sequences which led to reduce the data size, hence the required runtime. This technique also served to eliminate the randomness used in selecting potential motifs and to reduce the number of computations.
5. A bespoke pruning technique involving clustering, which further pruned infrequent sub-sequences from the data. This technique was also used for randomness elimination and computation reduction.
6. A selection technique for computation reduction based on "best of the best" idea, where two selection thresholds, max and k, were applied to a set of potential motifs obtained from previously pruned data.
7. Conflict resolution methods to address the issue whereby unlabelled time series to be classified using the SGR-FMD approach ended up with a number of class-labels, where one was required. The three methods considered to select the most appropriate class-label were: Shortest Distance, Shortest Total Distances and Highest Votes.
8. The process termination "early abandonment" concept also used in the previous two proposed approaches.

The SGR-FMD approach served to significantly reduce the processing time (by a factor of more than 278) compared to the previous approach, the PCG_{seg} approach. This is then the third contribution of the thesis and is discussed in further detail in Chapter 7.

- **Contribution 4, The CE-FCS Approach:** This approach was designed to address both motif quality and the required processing time by generating meaningful motifs and, at the same time, reducing the number of computations. This was achieved using the following techniques:

1. A "motif generator" technique, the PCG_{ce} technique, for extracting meaningful motifs, which was designed to isolate heartbeat cycles. A fundamental feature in this generator was that it considered all data points in time series in the generation process.
2. The frequency-based representation strategy.
3. The trivial-match exclusion technique.
4. The process termination using "early abandonment" concept.
5. The local and global representativeness idea.
6. The "zero-motif" pruning technique.
7. A second pruning technique that considered the statistical distribution of data to exclude irrelevant sub-sequences, which made the motif discovery process tractable.
8. The "best of the best" selection technique.
9. The conflict resolution methods from the previous approach.

The CE-FCS approach was found to be the best of the four presented approaches; its performance was compared to all the approaches proposed in this thesis as discussed in detail in Chapter 8. This is then the fourth, and last, contribution of the thesis.

9.4 Future Work

This final section considers some suggested directions for future work whereby the work presented in this thesis can be extended. Below is an itemisation of these potential future research directions:

- **Additional layer in the PCG_{seg} hierarchy:** In Chapter 6, a bespoke, two layer, PCG time series segmentation technique, PCG_{seg}, was designed to address the issue of the overall size of the PCG time series to be processed, which served to significantly reduce the processing time. However, it is suggested that another way of achieving this might be to add a third level into the hierarchy comprised of a sequences of segments (repetitive segments). Thus, the three layers, from bottom to top, would represent sub-segments, segments and a sequence of segments. For example, in Figure 6.2(a), Chapter 6, the time series comprises 31 units (segments), it could be compressed to be only 7 units, where the series of repetitive "vertical" and "slant" shapes represent a "zigzag" in the third level. The assumption here is that this might further shrink the data size dramatically, but still preserve the salient information, hence enhancing the efficiency and effectiveness of classification model generation and usage.

- **Hybrid approach:** In Chapters 7 and 8, two different pre-processing techniques were presented: (i) silent gap removal, PCG_{sgr}, and (ii) cycle extraction, PCG_{ce}. Each of which showed a positive effect on the outcome results. Therefore, it is thought that a combination of the two can provide for additional efficiency gains. This could be implemented by extracting the cycles and then removing the outer silent gaps; thus, an investigation of using the proposed FCS algorithm (Chapter 8) with the pre-processed data would be another useful direction for future work.

- **Additional data pre-processing:** The work described in this thesis used raw data collected by electronic stethoscopes. Although good experimental results were reported using this data, some additional pre-processing might produce further improvements. It is conjectured that smoothing and/or filtering the data may contribute to classification effectiveness as it will serve to remove noise. Removing less important parts of the data, such as noise, might help to only process the effective portions of the data, hence further reducing the required processing time.

- **Alternative similarity measurement:** The Euclidean Distance function was used throughout the work presented in this thesis as a similarity measurement because it is frequently used in the context of time series similarity calculation. Alternatively, Dynamic Time Warping (DTW) similarity measurement might be used. The advantage offered is that DTW is able to "warp" the linearity of time series, thus, it is able to capture similarity where offsets exist. This could then be fruitfully employed for motif similarity checking, specifically with respect to the CE-FCS approach where motifs are cycles, because the motifs can be "warped" to give a best similarity distance measurement. It is anticipated that this might improve overall classification performance.

- **Alternative domains:** One presented contribution, Section 9.3, was a novel method of reading WAVE files, where a time series point represents a Sample value in place of the usual Amplitude value. Further experiments demonstrated that this method can be applied to any WAVE file; thus in the context of alternative domains rather than Phonocardiogram classification. One such domain is human voice recording analysis, a requirement with respect to multiple applications, such as: authentication, translation and early psychiatric/physical diagnosis. The advantages of considering these fruitful topics with this method (reading a recording as Samples) is the reduction in data size to approximately half; hence reduced processing time.

Bibliography

[1] Btdigg. https://en.btdig.com/search?q=.wav+files, Last accessed on 01-04-2017.

[2] Hay bail wavs. http://thestablesbb.powweb.com/WAV%20FILES/wav_to_the_horse.htm, Last accessed on 01-04-2017.

[3] Uci machine learning repository, 1987. https://archive.ics.uci.edu/ml/index.php, Last accessed on 01-09-2019.

[4] Separation of biomedical signals, 2016. https://sisec.inria.fr/sisec-2016/bio2016/, Last accessed on 01-09-2019.

[5] NHS 5-year forward view, 2017. https://www.england.nhs.uk/five-year-forward-view/, Last accessed on 01-09-2019.

[6] Abbas K. Abbas and Rasha Bassam. Phonocardiography signal processing. 4:218, 04 2009.

[7] Ibrahim Abdel-Motaleb and Rohit Akula. Artificial intelligence algorithm for heart disease diagnosis using phonocardiogram signals. pages 1–6, 05 2012.

[8] M. Abo-Zahhad, Sabah M. Ahmed, and Sherif N. Abbas. Biometrics from heart sounds: Evaluation of a new approach based on wavelet packet cepstral features using hsct-11 database. *Computers & Electrical Engineering*, 53:346 – 358, 2016.

[9] Puneet Agarwal, Gautam Shroff, Sarmimala Saikia, and Zaigham Khan. Efficiently discovering frequent motifs in large-scale sensor data. In *Proceedings of the Second ACM IKDD Conference on Data Sciences (CoDS'15)*, pages 98–103, 2015.

[10] Rakesh Agrawal, Christos Faloutsos, and Arun Swami. *Efficient similarity search in sequence databases*, pages 69–84. Springer Berlin Heidelberg, Berlin, Heidelberg, 1993.

[11] Christer Ahlstrom. *NonLinear Phonocardiographic Signal Processing.* PhD thesis, Linkoping University, Sweden, April 2008.

[12] Hajar Alhijailan, Frans Coenen, Jo Dukes-McEwan, and Jeyarajan Thiyagalingam. Segmenting sound waves to support phonocardiogram analysis: The pcgseg approach. In Xin Geng and Byeong-Ho Kang, editors, *PRICAI 2018: Trends in Artificial Intelligence*, pages 100–112, Cham, 2018. Springer International Publishing.

[13] Andrés M. Alonso and Daniel Peña. Clustering time series by linear dependency. *Statistics and Computing*, 29(4):655–676, Jul 2019.

[14] Gennady Andrienko, Natalia Andrienko, Peter Bak, Sebastian Bremm, Daniel Keim, Tatiana von Landesberger, Christian Politz, and Tobias Schreck. A framework for using self-organising maps to analyse spatio-temporal patterns, exemplified by analysis of mobile phone usage. *Journal of Location Based Services*, 4(3-4):200–221, 2010.

[15] Rushil Anirudh and Pavan Turaga. Geometry-based symbolic approximation for fast sequence matching on manifolds. *International Journal of Computer Vision*, 116(2):161–173, Jan 2016.

[16] F. J. Anscombe. Graphs in statistical analysis. *The American Statistician*, 27(1):17–21, 1973.

[17] C. Atkins, J. Bonagura, S. Ettinger, P. Fox, S. Gordon, J. Haggstrom, R. Hamlin, B. Keene, V. Luis-Fuentes, and R. Stepien. Guidelines for the diagnosis and treatment of canine chronic valvular heart disease. *Journal of Veterinary Internal Medicine*, 23(6):1142–1150, 2009.

[18] Anthony Bagnall, Jon Hills, and Jason Lines. Finding motif sets in time series. *CoRR*, abs/1407.3685, 2014.

[19] D. Barschdorff, U. Femmer, and E. Trowitzsch. Automatic phonocardiogram signal analysis in infants based on wavelet transforms and artificial neural networks. In *Computers in Cardiology 1995*, pages 753–756, Sept 1995.

[20] Laura Beggel, Bernhard X. Kausler, Martin Schiegg, Michael Pfeiffer, and Bernd Bischl. Time series anomaly detection based on shapelet learning. *Computational Statistics*, 34(3):945–976, Sep 2019.

[21] Nurjahan Begum and Eamonn Keogh. Rare time series motif discovery from unbounded streams. *Proc. VLDB Endow.*, 8(2):149–160, oct 2014.

[22] Pooya Behroozinia, Seyedmeysam Khaleghian, Saied Taheri, and Reza Mirzaeifar. Damage diagnosis in intelligent tires using time-domain and frequency-domain analysis. *Mechanics Based Design of Structures and Machines*, 47(1):54–66, 2019.

[23] Ilya Belevich, Merja Joensuu, Darshan Kumar, Helena Vihinen, and Eija Jokitalo. Microscopy image browser: A platform for segmentation and analysis of multidimensional datasets. *PLOS Biology*, 14(1):1–13, 01 2016.

[24] John Bell. Life sciences industrial strategy - a report to the government from the life sciences sector, 2017.

[25] Peter Bentley, Glenn Nordehn, Miguel Coimbra, Shie Mannor, and Rita Getz. Classifying heart sounds challenge, 2011. `http://www.peterjbentley.com/heartchallenge/`, Last accessed on 01-09-2019.

[26] Laurent Besacier, Etienne Barnard, Alexey Karpov, and Tanja Schultz. Automatic speech recognition for under-resourced languages: A survey. *Speech Communication*, 56(Supplement C):85 – 100, 2014.

[27] Boris P Bezruchko and Dmitry A Smirnov. *Extracting Knowledge From Time Series: An Introduction to Nonlinear Empirical Modeling.* Springer Series in Synergetics. Springer-Verlag, 2013.

[28] Bosko Bozilovic, Branislav Todorovic, and Miroslav Obradovic. Text-independent speaker recognition using two-dimensional information entropy. *Journal of Electrical Engineering*, 66(3):169 – 173, 2015.

[29] O. Brandes, J. Farley, M. Hinich, and U. Zackrisson. The time domain and the frequency domain in time series analysis. *The Swedish Journal of Economics*, 70(1):25–42, 1968.

[30] Alan C. Braverman, Hasan Guven, Michael A. Beardslee, Majesh Makan, Andrew M. Kates, and Marc R. Moon. The bicuspid aortic valve. *Current Problems in Cardiology*, 30(9):470 – 522, 2005.

[31] Peter J Brockwell and Richard A Davis. *Introduction to Time Series and Forecasting.* Springer International Publishing, 3 edition, 2016.

[32] Kin-Pong Chan and Ada Fu. Efficient time series matching by wavelets. In *Proceedings - International Conference on Data Engineering*, pages 126–133, 04 1999.

[33] C H Chen, L F Pau, and P S P Wang. *Handbook of Pattern Recognition and Computer Vision.* World Scientific, 1993.

[34] T. Chen, S. Yang, L. Ho, K. Tsai, Y. Chen, Y. Chang, Y. Lai, S. Wang, Y. Tsao, and C. Wu. S1 and s2 heart sound recognition using deep neural networks. *IEEE Transactions on Biomedical Engineering*, 64(2):372–380, Feb 2017.

[35] L Hamza Cherif and SM Debba. Variability of pulmonary blood pressure, splitting of the second heart sound and heart rate. *Journal of Clinical & Experimental Cardiology*, 8(10):1–3, 2017.

[36] Bill Chiu, Eamonn Keogh, and Stefano Lonardi. Probabilistic discovery of time series motifs. In *Proceedings of the Ninth ACM SIGKDD International Conference*

on Knowledge Discovery and Data Mining, KDD '03, pages 493–498, New York, NY, USA, 2003. ACM.

[37] D. M. Cooper and E. F. Wood. Identifying multivariate time series models. *Journal of Time Series Analysis*, 3(3):153–164, May 1982.

[38] Belur V. Dasarathy. *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*. IEEE Computer Society Press tutorial. IEEE Computer Society Press, 1991. the University of Michigan.

[39] Hoang Anh Dau and Eamonn Keogh. Matrix profile v: A generic technique to incorporate domain knowledge into motif discovery. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 125–134, New York, NY, USA, 2017. ACM.

[40] Edilson Delgado-Trejos, A.F. Quiceno-Manrique, J.I. Godino-Llorente, M. Blanco-Velasco, and G. Castellanos-Dominguez. Digital auscultation analysis for heart murmur detection. *Annals of Biomedical Engineering*, 37(2):337–353, Feb 2009.

[41] Shi-Wen Deng and Ji-Qing Han. Towards heart sound classification without segmentation via autocorrelation feature and diffusion maps. *Future Generation Computer Systems*, 60:13 – 21, 2016.

[42] Hui Ding, Goce Trajcevski, Peter Scheuermann, Xiaoyue Wang, and Eamonn Keogh. Querying and mining of time series data: Experimental comparison of representations and distance measures. *Proc. VLDB Endow.*, 1(2):1542–1552, August 2008.

[43] Zumray Dokur and Tamer Olmez. Heart sound classification using wavelet transform and incremental self-organizing map. *Digital Signal Processing*, 18(6):951 – 959, 2008.

[44] Boubacar Doucoure, Kodjo Agbossou, and Alben Cardenas. Time series prediction using artificial wavelet neural network and multi-resolution analysis: Application to wind speed data. *Renewable Energy*, 92:202 – 211, 2016.

[45] L.G. Durand and Philippe Pibarot. A digital signal processing of the phonocardiogram: Review of the most recent advancements. 23:163–219, 02 1995.

[46] eGeneral Medical Inc. Cardiac auscultation of heart murmurs, 2000. `http://www.egeneralmedical.com/listohearmur.html`, Last accessed on 01-09-2019.

[47] Public Health England. Health matters: preventing cardiovascular disease, 2019. `https://www.gov.uk/government/publications/health-matters-preventing-cardiovascular-disease/health-matters-preventing-cardiovascular-disease`, Last accessed on 01-08-2019.

[48] Tasos Fratzolas. Find the perfect sound. `https://www.soundsnap.com`, Last accessed on 01-04-2017.

[49] Hamza Frihia and Halima Bahi. Hmm/svm segmentation and labelling of arabic speech for speech recognition applications. *International Journal of Speech Technology*, 20(3):563–573, Sep 2017.

[50] Yifeng Gao, Jessica Lin, and Huzefa Rangwala. Iterative grammar-based framework for discovering variable-length time series motifs. In *IEEE International Conference on Data Mining*, pages 111–116. IEEE, 11 2017.

[51] Ana Gavrovska, Milorad Paskas, Dragi Dujkovi, and Irini Reljin. Region-based phonocardiogram event segmentation in spectrogram image. In *10th Symposium on Neural Network Applications in Electrical Engineering*, pages 69–72, Sept 2010.

[52] Xianping Ge and Padhraic Smyth. Deformable markov model templates for time-series pattern matching. In *Proceeding of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1–18, 11 2002.

[53] Arash Gharehbaghi, Thierry Dutoit, Amir A. Sepehri, Armen Kocharian, and Maria Lindén. A novel method for screening children with isolated bicuspid aortic valve. *Cardiovascular Engineering and Technology*, 6(4):546–556, Dec 2015.

[54] Dawid Gradolewski, Giovanni Magenes, Sven Johansson, and Wlodek Kulesza. A wavelet transform-based neural network denoising algorithm for mobile phonocardiography. *Sensors*, 19(4):957, 2019.

[55] Yann-Gael Gueheneuc and Giuliano Antoniol. Demima: A multilayered approach for design pattern identification. *IEEE Transactions on Software Engineering*, 34(5):667–684, Sept 2008.

[56] Dame Wendy Hall and Jerome Pesenti. Growing the artificial intelligence industry in the uk, 2017.

[57] James D. Hamilton. *Time series analysis*. Levant Books, 2012.

[58] Lotfi Hamza Cherif, S. M. Debbal, and F. Bereksi-Reguig. Segmentation of heart sounds and heart murmurs. *Journal of Mechanics in Medicine and Biology*, 8(4):549–559, 2008.

[59] Jiawei Han, Jian Pei, and Micheline Kamber. *Data Mining: Concepts and Techniques*. The Morgan Kaufmann Series in Data Management Systems. Elsevier Science, 2011.

[60] E. J. Hannan. *Time series analysis*. Chapman and Hall, 1960.

[61] F. Hasfjord. Heart sound analysis with time dependent fractal dimensions. Master's thesis, Linkoping University, Sweden, 2004.

[62] Xiaoxu He, Chenxi Shao, and Yan Xiong. A non-parametric symbolic approximate representation for long time series. *Pattern Analysis and Applications*, 19(1):111–127, Feb 2016.

[63] Dora B. Heras, José Carlos Cabaleiro, Vicente Blanco, Pablo Costas, and Francisco F. Rivera. Principal component analysis on vector computers. In José M. L. M. Palma and Jack Dongarra, editors, *Vector and Parallel Processing — VECPAR'96*, pages 416–428, Berlin, Heidelberg, 1997. Springer Berlin Heidelberg.

[64] J. Leith Holloway. Smoothing and filtering of time series and space fields. In H.E. Landsberg and J. Van Mieghem, editors, *Advances in Geophysics*, volume 4, pages 351 – 389. Elsevier, 1958.

[65] J. R. M. Hosking. Lagrange-multiplier tests of multivariate time-series models. *Journal of the Royal Statistical Society. Series B (Methodological)*, 43(2):219–230, 1981.

[66] G. Huang and X. Zhou. A piecewise linear representation method of hydrological time series based on curve feature. In *2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC)*, volume 02, pages 203–207, Aug 2016.

[67] Kun-Yi Huang, Chung-Hsien Wu, Ming-Hsiang Su, and Yu-Ting Kuo. Detecting unipolar and bipolar depressive disorders from elicited speech responses using latent affective structure model. *IEEE Transactions on Affective Computing*, pages 1–13, 2018.

[68] Lucie N. Hutchins, Sean M. Murphy, Priyam Singh, and Joel H. Graber. Position-dependent motif characterization using non-negative matrix factorization. *Bioinformatics*, 24(23):2684–2690, 2008.

[69] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. Deep learning for time series classification: a review. *Data Mining and Knowledge Discovery*, 33(4):917–963, Jul 2019.

[70] Kyoung jae Kim. Financial time series forecasting using support vector machines. *Neurocomputing*, 55(1):307 – 319, 2003. Support Vector Machines.

[71] Puneet Kumar Jain and Anil Kumar Tiwari. An adaptive thresholding method for the wavelet based denoising of phonocardiogram signal. *Biomedical Signal Processing and Control*, 38:388 – 399, 2017.

[72] C. Ji, S. Liu, C. Yang, L. Wu, L. Pan, and X. Meng. A piecewise linear representation method based on importance data points for time series data. In *2016 IEEE*

20th International Conference on Computer Supported Cooperative Work in Design (CSCWD), pages 111–116, May 2016.

[73] Konstantinos Kalpakis, Dhiral Gada, and Vasundhara Puttagunta. Distance measures for effective clustering of arima time-series. In *Proceedings of the 2001 IEEE International Conference on Data Mining*, ICDM '01, pages 273–280, Washington, DC, USA, 2001. IEEE Computer Society.

[74] Norhaslinda Kamaruddin, Abdul W. A. Rahman, Khairul I. M. Halim, and Muhammad H. I. M. Noh. Driver behaviour state recognition based on speech. *TELKOMNIKA*, 16(2):852–861, 04 2018. Copyright - Copyright Ahmad Dahlan University Apr 2018; Last updated - 2018-04-27; SubjectsTermNotLitGenreText - United States–US; United Kingdom–UK; Malaysia; California.

[75] Wen-Chung Kao and Chih-Chao Wei. Automatic phonocardiograph signal analysis for detecting heart valve disorders. *Expert Systems with Applications*, 38(6):6458 – 6468, 2011.

[76] Steven Karban, James Drexler, and Cleon Hennen. Real time speech compaction/relay with silence detection, March 1983. Sperry Corp. US Patent 4,376,874.

[77] E Keogh, K Chakrabarti, and M Pazzani. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, 3:263–286, 01 2001.

[78] Eamonn Keogh, Kaushik Chakrabarti, Sharad Mehrotra, and Michael Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. In *ACM SIGMOD Record*, volume 30, pages 151–162, 06 2001.

[79] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and Information Systems*, 3(3):263–286, Aug 2001.

[80] Eamonn Keogh, Selina Chu, David Hart, and Michael Pazzani. Segmenting time series: A survey and novel approach. *Data Mining in Time Series Databases*, 57:1–22, 03 2003.

[81] Eamonn J. Keogh and Michael J. Pazzani. Derivative dynamic time warping. In *Proceedings of the 2001 SIAM International Conference on Data Mining*, pages 1–11.

[82] Eamonn J. Keogh and Michael J. Pazzani. An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In *Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining*, KDD'98, pages 239–243. AAAI Press, 1998.

[83] W. Khan and K. Kuru. An intelligent system for spoken term detection that uses belief combination. *IEEE Intelligent Systems*, 32(1):70–79, Jan 2017.

[84] Kazunori Komatani, Naoki Hotta, and Satoshi Sato. *Restoring Incorrectly Segmented Keywords and Turn-Taking Caused by Short Pauses*, pages 205–216. Springer International Publishing, Cham, 2016.

[85] Adam Krejci, Ted R. Hupp, Matej Lexa, Borivoj Vojtesek, and Petr Muller. Hammock: a hidden markov model-based peptide clustering algorithm to identify protein-interaction consensus motifs in large datasets. *Bioinformatics*, 32(1):9–16, January 2016.

[86] Martin Krzywinski and Naomi Altman. Significance, p values and t-tests. *Nature Methods*, 10(11):1041–1042, 11 2013.

[87] Ludmila I. Kuncheva. *Combining Pattern Classifiers: Methods and Algorithms.* John Wiley and Sons, Inc, 2nd ed. edition, 2014.

[88] Clifford Lam and Qiwei Yao. Factor modeling for high-dimensional time series: Inference for the number of factors. *Ann. Statist.*, 40(2):694–726, 04 2012.

[89] S. Latif, M. Usman, R. Rana, and J. Qadir. Phonocardiographic sensing using deep learning for abnormal heartbeat detection. *IEEE Sensors Journal*, 18(22):9393–9400, Nov 2018.

[90] Li-Wei Lee, Li-Hui Wang, and Shyi-Ming Chen. Temperature prediction and taifex forecasting based on high-order fuzzy logical relationships and genetic simulated annealing techniques. *Expert Systems with Applications*, 34(1):328 – 336, 2008.

[91] R. J. Lehner and R. M. Rangayyan. A three-channel microcomputer system for segmentation and characterization of the phonocardiogram. *IEEE Transactions on Biomedical Engineering*, BME-34(6):485–489, June 1987.

[92] Na Li, Martin Crane, Cathal Gurrin, and Heather J. Ruskin. Finding motifs in large personal lifelogs. In *Proceedings of the 7th Augmented Human International Conference 2016*, AH '16, pages 9:1–9:8, New York, NY, USA, 2016. ACM.

[93] H. Liang, S. Lukkarinen, and I. Hartimo. Heart sound segmentation algorithm based on heart sound envelogram. In *Computers in Cardiology 1997*, pages 105–108, Sep 1997.

[94] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Bill Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, DMKD '03, pages 2–11, New York, NY, USA, 2003. ACM.

[95] Jessica Lin, Eamonn Keogh, Stefano Lonardi, and Pranav Patel. Finding motifs in time series. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 53–68, 2002.

[96] Jessica Lin, Eamonn Keogh, Li Wei, and Stefano Lonardi. Experiencing sax: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15(2):107–144, Oct 2007.

[97] Jason Lines and Anthony Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565 – 592, 2015.

[98] Mbiadoun Lionel and martin kom. Remote tool for heart control and diagnostic through phonocardiography signal. *International Journal of Innovative Research in Science, Engineering and Technology*, 6:8, 04 2017.

[99] Chengyu Liu, David Springer, Qiao Li, Benjamin Moody, Ricardo Abad Juan, Francisco J Chorro, Francisco Castells, Jose Millet Roig, Ikaro Silva, Alistair E W Johnson, Zeeshan Syed, Samuel E Schmidt, Chrysa D Papadaniil, Leontios Hadjileontiadis, Hosein Naseri, Ali Moukadem, Alain Dieterlen, Christian Brandt, Hong Tang, Maryam Samieinasab, Mohammad Reza Samieinasab, Reza Sameni, Roger G Mark, and Gari D Clifford. An open access database for the evaluation of heart sound algorithms. *Physiological measurement*, 37(12):2181–2213, December 2016.

[100] Antoine Liutkus, Fabian-Robert Stöter, Zafar Rafii, Daichi Kitamura, Bertrand Rivet, Nobutaka Ito, Nobutaka Ono, and Julie Fontecave Jallon. The 2016 signal separation evaluation campaign. In Petr Tichavský, Massoud Babaie-Zadeh, Olivier J.J. Michel, and Nadège Thirion-Moreau, editors, *Latent Variable Analysis and Signal Separation - 12th International Conference, Liberec, Czech Republic, August 25-28, 2015, Proceedings*, pages 323–332. Springer International Publishing, 2017.

[101] Rangaraj M. Rangayyan and Richard J. Lehner. Phonocardiogram signal analysis: A review. 15:211–36, 02 1987.

[102] Shahid Ismail Malik and Imran Siddiqi. *Classification of Normal Heart Beats Using Spectral and Nonspectral Features for Phonocardiography Signals*, pages 13–24. Springer International Publishing, Cham, 2019.

[103] Miro Mannino and Azza Abouzied. Qetch: Time series querying with expressive sketches. In *Proceedings of the 2018 International Conference on Management of Data*, SIGMOD '18, pages 1741–1744, New York, NY, USA, 2018. ACM.

[104] Kenichiro McAlinn and Mike West. Dynamic bayesian predictive synthesis in time series forecasting. *Journal of Econometrics*, 210(1):155 – 169, 2019.

[105] McIntyre McIntyre. Ai and big data analytics: Driving more personalised health care. *Healthcare Tech Outlook*, 2017.

[106] Sheila R Messer, John Agzarian, and Derek Abbott. Optimal wavelet denoising for phonocardiograms. *Microelectronics Journal*, 32(12):931 – 941, 2001.

[107] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: Simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[108] Ali Moukadem, Alain Dieterlen, Nicolas Hueber, Chrisitian Brandt, and P. Raymond. Comparative study of heart sound localization algorithms. In Angel B. Rodriguez-Vazquez and Rainer Adelung, editors, *Proceedings of the SPIE, Bioelectronics, Biomedical, and Bioinspired Systems*, volume 8068, pages 1–9, Prague, May 2011. SPIE.

[109] Ali Moukadem, Alain Dieterlen, Nicolas Hueber, and Christian Brandt. A robust heart sounds segmentation module based on s-transform. *Biomedical Signal Processing and Control*, 8(3):273 – 281, 2013.

[110] Qurat-ul-ain Mubarak, Muhammad Usman Akram, Arslan Shaukat, and Aneeqa Ramazan. *Quality Assessment and Classification of Heart Sounds Using PCG Signals*, pages 1–11. Springer International Publishing, Cham, 2019.

[111] Abdullah Mueen. Time series motif discovery: dimensions and applications. 4(2):152–159, 03 2014.

[112] Abdullah Mueen, Eamonn Keogh, Qiang Zhu, Sydney Cash, and Brandon Westover. Exact discovery of time series motifs. In *Proceedings of the 2009 SIAM International Conference on Data Mining*, pages 473–484, 2009.

[113] Mohammed Nabih-Ali, EL-Sayed A. El-Dahshan, and Ashraf S. Yahia. Heart diseases diagnosis using intelligent algorithm based on pcg signal analysis. *International Journal of Biology and Biomedicine*, 2:81–85, 2017.

[114] Mohammed Nabih-Ali, EL-Sayed A. El-Dahshan, and Ashraf S. Yahia. A review of intelligent systems for heart sound signal analysis. *Journal of Medical Engineering & Technology*, 41(7):553–563, 2017.

[115] K. Nakamura, Shinya Kawamoto, Tatsuyuki Osuga, T. Morita, Nobuhiro Sasaki, K Morishita, Hitoya Ohta, and Mai Takiguchi. Left atrial strain at different stages of myxomatous mitral valve disease in dogs. *Journal of veterinary internal medicine*, 31(2):316–325, 2017.

[116] M. Nassralla, Z. E. Zein, and H. Hajj. Classification of normal and abnormal heart sounds. In *2017 Fourth International Conference on Advances in Biomedical Engineering*, pages 1–4, Oct 2017.

[117] N. Neelima, E. Sreenivasa Reddy, and N. Kalpitha. An efficient qbir system using adaptive segmentation and multiple features. *Procedia Computer Science*, 87(Supplement C):134 – 139, 2016. Fourth International Conference on Recent Trends in Computer Science & Engineering (ICRTCSE 2016).

[118] N. Obin, F. Lamare, and A. Roebel. Syll-o-matic: An adaptive time-frequency representation for the automatic segmentation of speech into syllables. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6699–6703, May 2013.

[119] The University of Michigan Health System. Heart sound & murmur library. `www.med.umich.edu/lrc/psb/heartsounds/index.htm`, Last accessed on 01-09-2019.

[120] J. Oliveira, C. Sousa, and M. Coimbra. Coupled hidden markov model for automatic ecg and pcg segmentation. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1023–1027, March 2017.

[121] Tamer Olmez and Zumray Dokur. Classification of heart sounds using an artificial neural network. *Pattern Recognition Letters*, 24(1):617 – 629, 2003.

[122] T. Oskiper and R. Watrous. Detection of the first heart sound using a time-delay neural network. In *Computers in Cardiology*, volume 29, pages 537–540, Sep. 2002.

[123] Soumya Priyadarsini Panda and Ajit Kumar Nayak. Automatic speech segmentation in syllable centric speech recognition system. *International Journal of Speech Technology*, 19(1):9–18, Mar 2016.

[124] Achilles J. Pappano and Withrow Gil Wier. Automaticity: Natural excitation of the heart. In Achilles J. Pappano and Withrow Gil Wier, editors, *Cardiovascular Physiology*, pages 31 – 53. Content Repository Only!, Philadelphia, tenth edition, 2013.

[125] Sanghyun Park, Dongwon Lee, and Wesley W. Chu. Fast retrieval of similar subsequences in long sequence databases. In *Proceedings of the 1999 Workshop on Knowledge and Data Engineering Exchange*, KDEX '99, pages 60–, Washington, DC, USA, 1999. IEEE Computer Society.

[126] Ameera X. Patel and Edward T. Bullmore. A wavelet-based estimator of the degrees of freedom in denoised fmri time series for probabilistic testing of functional connectivity and brain graphs. *NeuroImage*, 142(Supplement C):14 – 26, 2016.

[127] G. Peng, G. Zhou, D. T. Nguyen, X. Qi, Q. Yang, and S. Wang. Continuous authentication with touch behavioral biometrics and voice on wearable glasses. *IEEE Transactions on Human-Machine Systems*, 47(3):404–416, June 2017.

[128] D. Pham, S. Meignen, N. Dia, J. Fontecave-Jallon, and B. Rivet. Phonocardiogram signal denoising based on nonnegative matrix factorization and adaptive contour representation computation. *IEEE Signal Processing Letters*, 25(10):1475–1479, Oct 2018.

[129] Koksoon Phua, Jianfeng Chen, Tran Huy Dat, and Louis Shue. Heart sound as a biometric. *Pattern Recognition*, 41(3):906 – 919, 2008. Part Special issue: Feature Generation and Machine Learning for Robust Multimodal Biometrics.

[130] Bentley PM, Grant PM, and McDonnell JT. Time-frequency and time-scale techniques for the classification of native and bioprosthetic heart valve sounds. *IEEE Trans Biomed Eng.*, 45(1):125 – 132, 1998.

[131] Andy Pole, Mike West, and Jeff Harrison. *Applied Bayesian Forecasting and Time Series Analysis*. Chapman and Hall/CRC, New York, 1994.

[132] C. Potes, S. Parvaneh, A. Rahman, and B. Conroy. Ensemble of feature-based and deep learning-based classifiers for detection of abnormal heart sounds. In *2016 Computing in Cardiology Conference (CinC)*, pages 621–624, Sep. 2016.

[133] Xingri QUAN, Jongwon SEOK, and Keunsung BAE. Detection of s1/s2 components with extraction of murmurs from phonocardiogram. *IEICE Transactions on Information and Systems*, E98.D(3):745–748, 2015.

[134] A. F. Quiceno, E. Delgado, M. Vallverd, A. M. Matijasevic, and G. Castellanos-Domnguez. Effective phonocardiogram segmentation using nonlinear dynamic analysis and high-frequency decomposition. In *2008 Computers in Cardiology*, pages 161–164, Sept 2008.

[135] A.F. Quiceno-Manrique, J.I. Godino-Llorente, M. Blanco-Velasco, and G. Castellanos-Dominguez. Selection of dynamic features based on time-frequency representations for heart murmur detection from phonocardiographic signals. *Ann Biomed Eng*, 38(1):118 – 137, 2010.

[136] M R Homaeinezhad, Pouya Sabetan, Amir Feizollahi, Ali Ghaffari, and Rabin Rahmani. Parametric modelling of cardiac system multiple measurement signals: An open-source computer framework for performance evaluation of ecg, pcg and abp event detectors. 36:117–34, 02 2012.

[137] Sreeraman Rajan, R Doraiswami, R Stevenson, and Raymond Watrous. Wavelet based bank of correlators approach for phonocardiogram signal classification. pages 77 – 80, 11 1998.

[138] Javier Ramirez, Jose Segura, Carmen Benitez, Angel de la Torre, and Antonio Rubio. Efficient voice activity detection algorithms using long-term speech information. *Speech Communication*, 42(3):271 – 287, 2004.

[139] D.A. Ramli, M.Y. Hooi, and K.J. Chee. Development of heartbeat detection kit for biometric authentication system. *Procedia Computer Science*, 96:305 – 314, 2016. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 20th International Conference KES-2016.

[140] G.C. Reinsel. *Elements of Multivariate Time Series Analysis*. Springer Series in Statistics. Springer New York, 2003.

[141] Sodabeh Salehi Rekavandi, Hamidreza Ghaffary, and Maryam Davodpour. Recognition of speech isolated words based on pyramid phonetic bag of words model display and kernel-based support vector machine classifier model. In Shahram Montaser Kouhsari, editor, *Fundamental Research in Electrical Engineering*, pages 15–30, Singapore, 2019. Springer Singapore.

[142] M. Rouhani and R. Abdoli. A comparison of different feature extraction methods for diagnosis of valvular heart diseases using pcg signals. *Journal of Medical Engineering & Technology*, 36(1):42–49, 2012.

[143] Mohamed Rouis, Abdelkrim Ouafi, and Salim Sbaa. Optimal level and order detection in wavelet decomposition for pcg signal denoising. *Biomedical Engineering*, 64(2):163 – 176, 2018.

[144] Jonathan Rubin, Rui Abreu, Anurag Ganguli, Saigopal Nelaturi, Ion Matei, and Kumar Sricharan. Recognizing abnormal heart sounds using deep learning. *arXiv e-prints*, jul 2017.

[145] Pavlopoulos S, Stasis A, and Loukis E. A decision tree-based method for the differential diagnosis of aortic stenosis from mitral regurgitation using heart sounds. *Biomed Eng Online*, 3(1), 2004.

[146] Sumathi Sai and S.N. Sivanandam. Data mining in customer value and customer relationship management. 29:321–386, 01 2007.

[147] Ridvan Saracoglu. Hidden markov model-based classification of heart valve disease with pca for dimension reduction. *Engineering Applications of Artificial Intelligence*, 25(7):1523 – 1528, 2012. Advanced issues in Artificial Intelligence and Pattern Recognition for Intelligent Surveillance System in Smart Home Environment.

[148] F. Sattar, F. Jin, A. Moukadem, C. Brandt, and A. Dieterlen. *Time-Scale-Based Segmentation for Degraded PCG Signals Using NMF*, pages 179–194. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[149] Alicia N. Schep, Beijing Wu, Jason D. Buenrostro, and William J. Greenleaf. chromvar: inferring transcription-factor-associated accessibility from single-cell epigenomic data. *Nature Methods*, 14:975 – 978, Aug 2017.

[150] Joan Serrà and Josep Lluís Arcos. Particle swarm optimization for time series motif discovery. *CoRR*, abs/1501.07399, 2015.

[151] Mehdi Sharifzadeh, Farnaz Azmoodeh, and Cyrus Shahabi. *Change Detection in Time Series Data Using Wavelet Footprints*, pages 127–144. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[152] H. Shatkay and S. B. Zdonik. Approximate queries and representations for large data sequences. In *Proceedings of the Twelfth International Conference on Data Engineering*, pages 536–545, New Orleans, LA, USA, Feb 1996. IEEE.

[153] Jack Sklansky and Victor Gonzalez. Fast polygonal approximation of digitized curves. *Pattern Recognition*, 12(5):327 – 331, 1980.

[154] Jongseo Sohn, Nam Soo Kim, and Wonyong Sung. A statistical model-based voice activity detection. *IEEE Signal Processing Letters*, 6(1):1 – 3, Jan 1999.

[155] Marina Sokolova and Guy Lapalme. A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, 45(4):427 – 437, 2009.

[156] Qiang Song and Brad S. Chissom. Forecasting enrollments with fuzzy time series - part ii. *Fuzzy Sets and Systems*, 62(1):1 – 8, 1994.

[157] Andrea Spadaccini and Francesco Beritelli. Heart sounds catania 2011, 2011. `http://www.diit.unict.it/hsct11/`, Last accessed on 01-09-2019.

[158] Andrea Spadaccini and Francesco Beritelli. Performance evaluation of heart sounds biometric systems on an open dataset. In *2013 18th International Conference on Digital Signal Processing (DSP)*, pages 1–5, Fira, Greece, July 2013. IEEE.

[159] D. B. Springer, L. Tarassenko, and G. D. Clifford. Logistic regression-hsmm-based heart sound segmentation. *IEEE Transactions on Biomedical Engineering*, 63(4):822–832, April 2016.

[160] Milos B. Stojanovic, Milos M. Bozic, Milena M. Stankovic, and Zoran P. Stajic. A methodology for training set instance selection using mutual information in time series prediction. *Neurocomputing*, 141(Supplement C):236 – 245, 2014.

[161] D. Stowell, D. Giannoulis, E. Benetos, M. Lagrange, and M. D. Plumbley. Detection and classification of acoustic scenes and events. *IEEE Transactions on Multimedia*, 17(10):1733–1746, Oct 2015.

[162] Zbigniew Struzik and Arno Siebes. Measuring time series' similarity through large singular features revealed with wavelet transformation. In *Physica A-statistical Mechanics and Its Applications - PHYSICA A*, pages 162–166, 01 1999.

[163] Supreeya Swarup and Amgad N Makaryus. Digital stethoscope: technology update. *Medical devices (Auckland, N.Z.)*, 11:29–36, January 2018.

[164] Zeeshan Hassan Syed. MIT Automated Auscultation System. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, Cambridge, USA, 2003.

[165] Yoshiki Tanaka, Kazuhisa Iwamoto, and Kuniaki Uehara. Discovery of time-series motif from multi-dimensional data based on mdl principle. *Machine Learning*, 58(2):269–300, Feb 2005.

[166] Hong Tang, Ziyin Dai, Yuanlin Jiang, Ting Li, and Chengyu Liu. Pcg classification using multidomain features and svm classifier. *BioMed Research International*, 2018:1–14, 2018.

[167] Jiliang Tang, Salem Alelyani, and Huan Liu. Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, 37:1–33, 2014.

[168] Morton E. Tavel and Hart Katz. Usefulness of a new sound spectral averaging technique to distinguish an innocent systolic murmur from that of aortic stenosis. *The American Journal of Cardiology*, 95(7):902 – 904, 2005.

[169] Gert Thijs, Kathleen Marchal, Magali Lescot, Stephane Rombauts, Bart De Moor, Pierre Rouze, and Yves Moreau. A gibbs sampling method to detect overrepresented motifs in the upstream regions of coexpressed genes. *Journal of Computational Biology*, 9(2):447–464, 04 2004.

[170] George C. Tiao and Ruey S. Tsay. Model specification in multivariate time series. *Journal of the Royal Statistical Society. Series B (Methodological)*, 51(2):157–213, 1989.

[171] Sahar Torkamani and Volker Lohweg. Survey on time series motif discovery. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 7(2):e1199–n/a, 2017. e1199 DMKD-00255.R2.

[172] Ruey S. Tsay. Identifying multivariate time series models. *Journal of Time Series Analysis*, 10(4):357–372, July 1989.

[173] Arijit Ukil, Soma Bandyopadhyay, Chetanya Puri, Rituraj Singh, and Arpan Pal. Effective noise removal and unified model of hybrid feature space optimization for automated cardiac anomaly detection using phonocardiogarm signals. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 866–870, April 2018.

[174] Shiraz University. Classification of normal/abnormal heart sound recordings: the physionet/computing in cardiology challenge 2016, 2016. `https://physionet.org/challenge/2016/`, Last accessed on 01-09-2019.

[175] Kalaivani V, Devi R. L, and Anusuyadevi V. Phonocardiographic signal and electrocardiographic signal analysis for the detection of cardiovascular diseases. *Biosciences Biotechnology Research Asia*, 15(1), 2018.

[176] Alireza Vahdatpour, Navid Amini, and Majid Sarrafzadeh. Toward unsupervised activity discovery using multi-dimensional motif detection in time series. In *Proceedings of the 21st International Jont Conference on Artifical Intelligence*, IJCAI'09, pages 1261–1266, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.

[177] Amar Vaswani, Hwan Juet Khaw, Scott Dougherty, Vipin Zamvar, and Chim Lang. *Cardiology in a Heartbeat.* Scion Publishing Limited, 2015.

[178] Eva Volna, Martin Kotyrba, and Michal Janosek. *Pattern Recognition and Classification in Time Series Data.* IGI Global, 2016.

[179] Agnel Waghela, Rohan Reddy, Shivangi Rai, Aditya Pawar, and Namrata Gharat. Suv detection algorithm for speech signals. *International Journal of Advanced Research in Computer Science and Software Engineering*, 4(4):958 – 963, April 2014.

[180] Changzhou Wang and Xiaoyang Sean Wang. Supporting content-based searches on time series via approximation. In *Proceedings of the 12th International Conference on Scientific and Statistical Database Management*, SSDBM '00, pages 69–81, Washington, DC, USA, 2000. IEEE Computer Society.

[181] Ping Wang, Chu Sing Lim, Sunita Chauhan, Jong Yong Foo, and Venkataraman Anantharaman. Phonocardiographic signal analysis method using a modified hidden markov model. *Annals of Biomedical Engineering*, 53(3):367 – 374, 2007.

[182] Xiaqing Wang, Zicheng Fang, Peng Wang, Ruiyuan Zhu, and Wei Wang. *A Distributed Multi-level Composite Index for KNN Processing on Long Time Series*, pages 215–230. Springer International Publishing, Cham, 2017.

[183] William W. S. Wei. *Time Series Analysis: univariate and multivariate methods, classic version.* PRENTICE HALL, 2018.

[184] Wikipedia. Wikimedia commons. `https://commons.wikimedia.org/wiki/Category:Audio_files_of_speeches`, Last accessed on 01-04-2017.

[185] C.L. Wu and K.W. Chau. Data-driven models for monthly streamflow time series prediction. *Engineering Applications of Artificial Intelligence*, 23(8):1350 – 1367, 2010.

[186] Qing Xie, Chaoyi Pang, Xiaofang Zhou, Xiangliang Zhang, and Ke Deng. Maximum error-bounded piecewise linear representation for online stream approximation. *The VLDB Journal*, 23(6):915–937, Dec 2014.

[187] A. Yadav, M. K. Dutta, C. M. Travieso, and J. B. Alonso. Automatic classification of normal and abnormal pcg recording heart sound recording using fourier transform. In *2018 IEEE International Work Conference on Bioinspired Intelligence (IWOBI)*, pages 1–9, July 2018.

[188] A. Yadollahi and Z. M. K. Moussavi. A robust method for heart sounds localization using lung sounds entropy. *IEEE Transactions on Biomedical Engineering*, 53(3):497–502, March 2006.

[189] D. Yang, H. Chen, Y. Song, and Z. Gong. Granger causality for multivariate time series classification. In *2017 IEEE International Conference on Big Knowledge (ICBK)*, pages 103–110, Aug 2017.

[190] Xiaoling Yang, Baohua Tan, Jiehua Ding, Jinye Zhang, and Jiaoli Gong. Comparative study on voice activity detection algorithm. In *Proceedings of the 2010 International Conference on Electrical and Control Engineering*, ICECE '10, pages 599–602, Washington, DC, USA, 2010. IEEE Computer Society.

[191] Byoung-Kee Yi and Christos Faloutsos. Fast time sequence indexing for arbitrary lp norms. In *Proceedings of the 26th International Conference on Very Large Data Bases, VLDB'00*, pages 385–394, 01 2000.

[192] Fatemeh Zare-Mirakabad, Hayedeh Ahrabian, MehdeiSadeghi, Abbas Nowzari-Dalini, and Bahram Goliaei. New scoring schema for finding motifs in dna sequences. *BMC bioinformatics*, 10(93), 03 2009.

[193] H. Zhao, Z. Dong, T. Li, X. Wang, and Pang C. Segmenting time series with connected lines under maximum error bound. *Information Sciences*, 345:1–8, 2016.

[194] H. Zhao, G. Li, H. Zhang, and Y. Xue. An improved algorithm for segmenting online time series with error bound guarantee. *Int. Jo. of Machine Learning and Cybernetics*, 7(3):365–374, 2016.

[195] Huan-yu Zhao, Guang-xia Li, Hao-lan Zhang, and Yun Xue. An improved algorithm for segmenting online time series with error bound guarantee. *International Journal of Machine Learning and Cybernetics*, 7(3):365–374, Jun 2016.

[196] Huanyu Zhao, Zhaowei Dong, Tongliang Li, Xizhao Wang, and Chaoyi Pang. Segmenting time series with connected lines under maximum error bound. *Information Sciences*, 345(Supplement C):1 – 8, 2016.

[197] Yanxia Zhao, Dingguo Xu1, Shouzhong Xiao, Xiaobo Yan, Jianming Liu, Yong Liu, Linmei Luo, and Guoxiang Xia. Measurement of two new indicators of cardiac reserve in humans, rats, rabbits, and dogs. *Journal of Biomedical Science and Engineering*, 6(10):960–963, 10 2013.

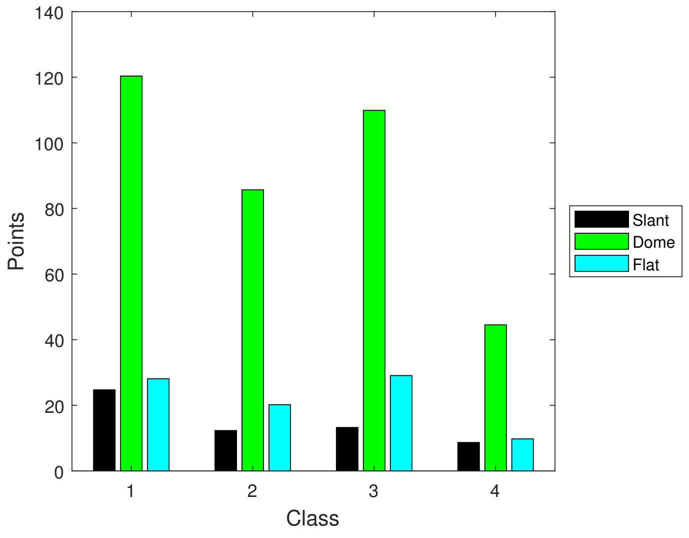
[198] Zhidong Zhao, Qinqin Shen, and Fangqin Ren. Heart sound biometric system based on marginal spectrum analysis. *Sensors*, 13(2):2530–2551, 2013.

Appendix A

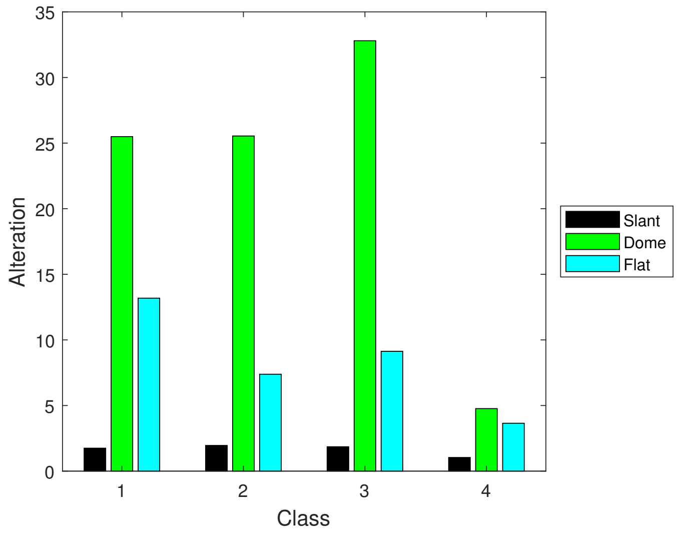
Additional Statistics Concerning the Segmentation Results for the PCG$_{\text{seg}}$ Classification Approach Presented in Chapter 6

A.1 Overview

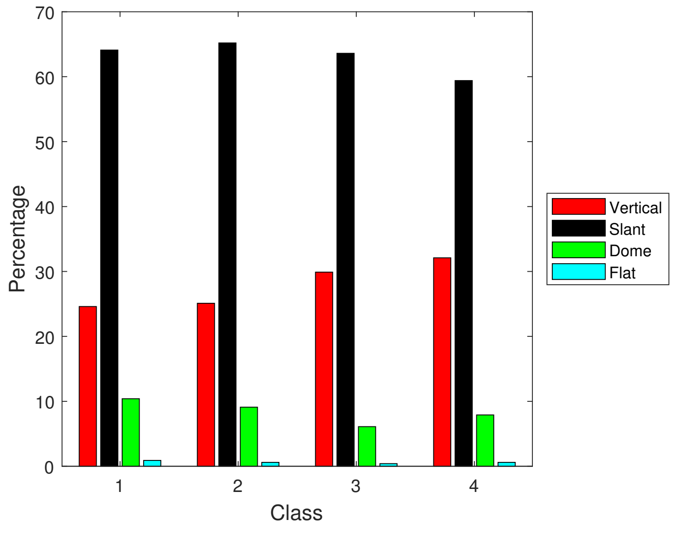
This appendix presents some additional results concerning the PCG$_{\text{seg}}$ Classification approach presented in Chapter 6. In particular, statistics related to the segmentation results using the PCG$_{\text{seg}}$ approach are presented in Figure A.1. Figure A.1(a) gives the average number of points in "slant", "dome" and "flat" shapes with respect to each one of the four classes in the PCG data set. Figure A.1(b) shows the average trend alterations in each segment; "trend alteration" in this context means the change in the sub-segment *type*. The "vertical" shape always comprises two points and is therefore not shown in Figure A.1(a). Similarly, it does not appear in Figure A.1(b), because there is only one trend in the "vertical" shape. Figure A.1(c) presents the average ratio of each segment to the PCG time series in terms of length.



(a) Average length of each segment



(b) Average trend alterations in each segment



(c) Average ratio of each segment to the time series in terms of length

Figure A.1: Sequence of plots illustrating some statistics concerning the segmentation proposed for the PCG$_{seg}$ approach

Appendix B

Further Results Concerning the SGR-FMD Approach Presented in Chapter 7

B.1 Overview

This appendix presents some further results concerning the experiments using the SGR-FMD approach presented in Chapter 7, where some one or two of the proposed pruning processes were omitted. The classification results of using all three pruning processes were presented earlier in Table 7.6, Chapter 7. In this appendix, the presented experiments were as follows:

- **Case 1:** Silent Gap Removal process omitted.
- **Case 2:** Candidate Frequent Motif Generation process omitted.
- **Case 3:** Both the Silent Gap Removal and the Candidate Frequent Motif Generation processes omitted.

The classification performance results with respect to these three cases are presented in Tables B.1, B.2 and B.3 respectively. The recorded runtime results for the three cases is presented in Table B.4.

Table B.1: Classification performance measures for the SGR-FMD approach (case 1)

CRM	ω	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	100	10	0.701	0.101	0.250	0.143	0.701	0.101	0.250	0.143
		20	0.708	0.130	0.275	0.173	0.708	0.130	0.275	0.173
		30	0.707	0.166	0.295	0.208	0.707	0.166	0.295	0.208
	200	10	0.687	0.126	0.237	0.161	0.709	0.204	0.320	0.245
		20	0.708	0.140	0.266	0.178	0.694	0.099	0.241	0.140
		30	0.694	0.099	0.241	0.139	0.694	0.099	0.241	0.139
	300	10	0.673	0.135	0.237	0.165	**0.716**	0.218	0.324	0.251
		20	0.707	0.160	0.283	0.200	0.708	0.213	0.329	0.253
		30	0.715	0.215	0.312	0.247	0.709	0.187	0.312	0.229
STD	100	10	0.693	0.129	0.258	0.169	0.693	0.129	0.258	0.169
		20	0.715	0.208	0.308	0.239	0.693	0.186	0.266	0.214
		30	0.693	0.179	0.308	0.222	0.700	0.187	0.312	0.228
	200	20	0.666	0.142	0.233	0.171	0.686	0.188	0.283	0.219
		20	0.678	0.166	0.253	0.195	0.672	0.106	0.220	0.141
		30	0.700	0.207	0.320	0.243	0.679	0.165	0.258	0.196
	300	10	0.659	0.256	0.299	0.271	0.709	0.229	0.316	0.259
		20	0.636	0.177	0.249	0.202	0.714	0.239	0.316	0.265
		30	0.708	0.244	0.333	0.276	0.686	0.162	0.257	0.194
HV	100	10	0.701	0.103	0.250	0.145	0.701	0.101	0.250	0.143
		20	0.715	0.168	0.304	0.210	0.711	0.209	0.271	0.214
		30	0.701	0.137	0.283	0.182	0.701	0.137	0.283	0.182
	200	10	0.687	0.172	0.258	0.196	0.702	0.179	0.287	0.211
		20	0.702	0.174	0.291	0.211	0.709	0.187	0.312	0.229
		30	0.695	0.141	0.274	0.183	0.702	0.143	0.283	0.187
	300	10	0.709	0.215	0.295	0.241	0.687	0.360	0.412	0.380
		20	0.660	0.262	0.308	0.270	0.686	0.320	0.378	0.331
		30	0.694	0.172	0.278	0.208	0.687	0.145	0.266	0.185

Table B.2: Classification performance measures for the SGR-FMD approach (case 2)

CRM	ω	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	100	10	0.664	0.237	0.307	0.264	0.678	0.249	0.324	0.276
		20	0.639	0.178	0.233	0.193	0.646	0.178	0.237	0.195
		30	0.619	0.164	0.183	0.166	0.626	0.176	0.191	0.178
	200	10	0.607	0.163	0.216	0.182	0.613	0.178	0.216	0.193
		20	0.664	0.239	0.349	0.276	0.665	0.237	0.362	0.275
		30	0.645	0.263	0.341	0.293	0.653	0.273	0.345	0.301
	300	10	0.579	0.112	0.108	0.107	0.574	0.099	0.108	0.100
		20	0.618	0.181	0.241	0.199	0.637	0.216	0.287	0.238
		30	0.678	0.322	0.354	0.322	0.652	0.266	0.308	0.275
STD	100	10	0.623	0.228	0.253	0.237	0.658	0.247	0.312	0.273
		20	0.567	0.141	0.123	0.123	0.596	0.183	0.249	0.190
		30	0.581	0.125	0.137	0.129	0.555	0.060	0.104	0.069
	200	10	0.627	0.183	0.299	0.215	0.620	0.181	0.249	0.205
		20	0.645	0.188	0.295	0.224	0.638	0.201	0.349	0.241
		30	0.661	0.311	0.324	0.311	0.626	0.195	0.253	0.201
	300	10	0.579	0.129	0.125	0.124	0.583	0.116	0.125	0.116
		20	0.624	0.179	0.224	0.194	0.641	0.185	0.291	0.219
		30	0.625	0.176	0.237	0.185	0.598	0.162	0.187	0.166
HV	100	10	0.680	0.286	0.374	0.313	0.665	0.274	0.358	0.300
		20	0.674	0.175	0.257	0.201	0.668	0.116	0.224	0.151
		30	**0.693**	0.164	0.278	0.201	0.660	0.262	0.308	0.270
	200	10	0.578	0.120	0.141	0.121	0.578	0.131	0.158	0.130
		20	0.679	0.279	0.395	0.315	0.665	0.274	0.349	0.301
		30	0.644	0.264	0.333	0.287	0.637	0.245	0.295	0.259
	300	10	0.662	0.175	0.183	0.176	0.599	0.091	0.095	0.089
		20	0.645	0.254	0.295	0.261	0.673	0.295	0.329	0.305
		30	0.652	0.260	0.304	0.267	0.670	0.301	0.358	0.311

Table B.3: Classification performance measures for the SGR-FMD approach (case 3)

CRM	ω	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	100	10	0.680	0.119	0.241	0.156	0.680	0.119	0.241	0.156
		20	0.679	0.096	0.224	0.133	0.679	0.096	0.224	0.133
		30	0.673	0.092	0.216	0.128	0.673	0.092	0.216	0.128
	200	10	0.678	0.322	0.354	0.322	0.701	0.152	0.283	0.194
		20	0.644	0.179	0.237	0.201	0.658	0.185	0.254	0.211
		30	0.694	0.099	0.241	0.139	0.694	0.098	0.241	0.138
	300	10	0.699	0.255	0.341	0.287	0.669	0.207	0.283	0.234
		20	0.673	0.135	0.237	0.165	0.700	0.164	0.274	0.199
		30	0.676	0.201	0.287	0.231	0.676	0.201	0.287	0.231
STD	100	10	0.629	0.133	0.208	0.158	0.636	0.135	0.220	0.163
		20	0.658	0.114	0.216	0.146	0.672	0.144	0.249	0.176
		30	0.693	0.164	0.278	0.201	0.666	0.142	0.233	0.171
	200	10	0.672	0.151	0.249	0.184	0.695	0.186	0.304	0.227
		20	0.694	0.241	0.316	0.268	0.618	0.181	0.241	0.199
		30	0.679	0.152	0.262	0.188	0.678	0.154	0.262	0.191
	300	10	0.652	0.195	0.245	0.211	0.695	0.200	0.287	0.232
		20	0.686	0.186	0.262	0.206	0.686	0.202	0.295	0.232
		30	0.680	0.170	0.287	0.211	0.688	0.175	0.287	0.210
HV	100	10	0.695	0.150	0.253	0.183	0.680	0.131	0.233	0.163
		20	0.701	0.124	0.254	0.162	0.695	0.141	0.274	0.183
		30	0.695	0.148	0.274	0.188	0.695	0.170	0.291	0.211
	200	10	0.702	0.179	0.287	0.211	0.695	0.138	0.249	0.170
		20	0.660	0.262	0.308	0.270	0.686	0.320	0.378	0.331
		30	0.680	0.122	0.241	0.159	0.680	0.124	0.241	0.160
	300	10	0.693	0.236	0.333	0.272	**0.703**	0.176	0.299	0.218
		20	0.695	0.199	0.308	0.238	0.702	0.181	0.241	0.199
		30	0.701	0.199	0.291	0.229	0.695	0.172	0.278	0.206

Table B.4: Runtime results (ss.SS format) for the SGR-FMD approach, where some pruning processes omitted

ω	k	Case 1	Case 2	Case 3
100	10	00.76	00.79	01.29
	20	01.32	06.30	10.78
	30	01.88	**15.39**	28.33
200	10	02.10	02.37	04.04
	20	04.09	09.43	16.21
	30	06.00	21.20	36.47
300	10	**03.74**	04.81	**08.27**
	20	07.13	14.22	24.50
	30	10.82	28.48	48.90
Average		04.20	11.44	19.87

Appendix C

Further Results Concerning the CE-FCS Approach Presented in Chapter 8

C.1 Overview

This appendix presents some additional results concerning the experiments using the CE-FCS approach presented in Chapter 8. The experiments used different values for the parameters: λ, σ_m and $flag$ as shown in Table C.1. This resulted in 10 tables presented below.

Table C.1: Structure of tables presented in this appendix

λ	σ_m	$flag$	Table
17e5	0.1	all	C.2
		max	C.3
91e5		all	C.4
		max	C.5
164e5		all	C.6
		max	C.7
238e5		all	C.8
		max	C.9
	1	all	C.10
		max	C.11

Table C.2: Classification performance measures for the CE-FCS approach with "all" pruning option using $\sigma_m = 0.1$ and $\lambda = 17e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.583	0.156	0.178	0.160	0.574	0.117	0.159	0.124
		20	0.566	0.147	0.138	0.135	0.591	0.190	0.194	0.181
		30	0.575	0.096	0.184	0.115	0.575	0.043	0.191	0.069
	2	10	0.617	0.155	0.270	0.172	0.609	0.104	0.260	0.130
		20	0.592	0.099	0.218	0.120	0.601	0.069	0.249	0.106
		30	0.583	0.096	0.193	0.105	0.583	0.048	0.208	0.077
	3	10	0.609	0.125	0.245	0.144	0.601	0.074	0.235	0.103
		20	0.583	0.068	0.193	0.092	0.608	0.119	0.251	0.142
		30	0.583	0.098	0.193	0.106	0.600	0.115	0.234	0.135
STD	1	10	0.609	0.261	0.208	0.225	0.617	0.221	0.240	0.208
		20	0.583	0.183	0.158	0.165	0.608	0.220	0.236	0.204
		30	0.583	0.152	0.203	0.158	0.591	0.191	0.203	0.188
	2	10	0.558	0.044	0.141	0.066	0.592	0.159	0.241	0.188
		20	0.549	0.086	0.118	0.091	0.584	0.180	0.178	0.156
		30	0.609	0.238	0.218	0.221	0.558	0.115	0.134	0.097
	3	10	0.566	0.077	0.174	0.105	0.558	0.063	0.141	0.084
		20	0.583	0.114	0.171	0.122	0.558	0.156	0.126	0.123
		30	0.583	0.117	0.136	0.122	0.583	0.224	0.164	0.175
HV	1	10	0.601	0.097	0.235	0.110	0.592	0.047	0.225	0.077
		20	0.601	0.050	0.250	0.082	0.592	0.046	0.225	0.075
		30	0.592	0.046	0.225	0.075	0.592	0.046	0.225	0.075
	2	10	0.592	0.132	0.183	0.137	0.576	0.068	0.163	0.091
		20	0.634	0.118	0.261	0.149	0.634	0.104	0.261	0.138
		30	0.610	0.077	0.260	0.112	0.601	0.050	0.250	0.082
	3	10	**0.667**	0.141	0.251	0.174	0.626	0.108	0.223	0.140
		20	0.617	0.079	0.238	0.115	0.642	0.137	0.271	0.162
		30	0.610	0.077	0.300	0.120	0.601	0.050	0.250	0.082

Table C.3: Classification performance measures for the CE-FCS approach with "max" pruning option using $\sigma_m = 0.1$ and $\lambda = 17e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.566	0.147	0.138	0.135	0.591	0.190	0.194	0.181
		20	0.567	0.068	0.168	0.094	0.567	0.039	0.174	0.064
		30	0.601	0.160	0.221	0.167	0.592	0.147	0.211	0.145
	2	10	0.592	0.099	0.218	0.120	0.601	0.069	0.249	0.106
		20	0.583	0.095	0.193	0.105	0.583	0.046	0.208	0.074
		30	0.583	0.070	0.193	0.094	0.583	0.045	0.208	0.073
	3	10	0.583	0.068	0.193	0.092	0.608	0.119	0.251	0.142
		20	0.583	0.098	0.193	0.107	0.583	0.048	0.208	0.077
		30	0.592	0.148	0.203	0.151	0.583	0.096	0.193	0.116
STD	1	10	0.583	0.183	0.158	0.165	0.608	0.220	0.236	0.204
		20	0.601	0.196	0.223	0.196	0.583	0.142	0.201	0.150
		30	0.601	0.206	0.239	0.216	0.619	0.222	0.279	0.237
	2	10	0.549	0.086	0.118	0.091	0.584	0.180	0.178	0.156
		20	0.592	0.213	0.198	0.197	0.575	0.119	0.161	0.121
		30	0.592	0.167	0.196	0.170	0.575	0.119	0.176	0.123
	3	10	0.583	0.114	0.171	0.122	0.558	0.156	0.126	0.123
		20	0.599	0.153	0.178	0.163	0.593	0.146	0.153	0.137
		30	0.625	0.211	0.244	0.223	0.592	0.148	0.191	0.154
HV	1	10	0.601	0.050	0.250	0.082	0.592	0.046	0.225	0.075
		20	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
		30	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
	2	10	0.634	0.118	0.261	0.149	0.634	0.104	0.261	0.138
		20	0.592	0.046	0.225	0.075	0.601	0.050	0.250	0.082
		30	0.592	0.046	0.225	0.075	0.601	0.051	0.250	0.084
	3	10	0.617	0.079	0.238	0.115	**0.642**	0.137	0.271	0.162
		20	0.592	0.046	0.225	0.076	0.601	0.051	0.250	0.084
		30	0.592	0.046	0.225	0.075	0.592	0.046	0.225	0.076

Table C.4: Classification performance measures for the CE-FCS approach with "all" pruning option using $\sigma_m = 0.1$ and $\lambda = 91e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.592	0.136	0.203	0.151	0.601	0.115	0.234	0.136
		20	0.583	0.113	0.193	0.130	0.575	0.066	0.183	0.094
		30	0.574	0.091	0.176	0.108	0.574	0.039	0.183	0.064
	2	10	0.617	0.155	0.270	0.172	0.609	0.104	0.260	0.130
		20	0.592	0.099	0.218	0.120	0.609	0.120	0.259	0.144
		30	0.583	0.096	0.193	0.105	0.592	0.076	0.218	0.108
	3	10	0.601	0.075	0.235	0.104	0.601	0.075	0.235	0.104
		20	0.592	0.073	0.210	0.099	0.592	0.073	0.210	0.099
		30	0.592	0.113	0.210	0.124	0.583	0.069	0.193	0.092
STD	1	10	0.583	0.160	0.178	0.160	0.558	0.057	0.124	0.074
		20	0.566	0.195	0.123	0.149	0.549	0.066	0.125	0.082
		30	0.558	0.130	0.145	0.128	0.583	0.147	0.234	0.156
	2	10	0.583	0.063	0.200	0.094	0.600	0.173	0.218	0.178
		20	0.575	0.063	0.174	0.090	0.566	0.100	0.149	0.113
		30	0.583	0.062	0.191	0.091	0.567	0.066	0.174	0.088
	3	10	0.575	0.098	0.184	0.125	0.610	0.213	0.229	0.200
		20	0.584	0.066	0.208	0.097	0.558	0.064	0.132	0.083
		30	0.567	0.054	0.166	0.078	0.567	0.104	0.158	0.114
HV	1	10	0.584	0.043	0.216	0.071	0.592	0.094	0.226	0.116
		20	0.601	0.050	0.250	0.082	0.592	0.046	0.233	0.077
		30	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
	2	10	0.635	0.168	0.283	0.207	0.584	0.081	0.188	0.108
		20	0.675	0.238	0.283	0.244	0.642	0.117	0.221	0.147
		30	**0.695**	0.225	0.321	0.260	0.675	0.167	0.261	0.192
	3	10	0.686	0.209	0.276	0.224	0.619	0.108	0.204	0.138
		20	0.670	0.142	0.235	0.167	0.686	0.174	0.276	0.202
		30	0.670	0.090	0.220	0.126	0.678	0.166	0.251	0.189

Table C.5: Classification performance measures for the CE-FCS approach with "max" pruning option using $\sigma_m = 0.1$ and $\lambda = 91e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.583	0.113	0.193	0.130	0.575	0.066	0.183	0.094
		20	0.583	0.073	0.201	0.103	0.583	0.045	0.208	0.073
		30	0.583	0.095	0.194	0.120	0.575	0.090	0.184	0.098
	2	10	0.592	0.099	0.218	0.120	0.609	0.120	0.259	0.144
		20	0.583	0.096	0.193	0.105	0.592	0.099	0.218	0.121
		30	0.583	0.072	0.193	0.096	0.592	0.066	0.224	0.100
	3	10	0.592	0.073	0.210	0.099	0.592	0.073	0.210	0.099
		20	0.583	0.097	0.193	0.106	0.583	0.070	0.193	0.095
		30	0.600	0.174	0.220	0.177	0.600	0.172	0.220	0.176
STD	1	10	0.566	0.195	0.123	0.149	0.549	0.066	0.125	0.082
		20	0.592	0.162	0.204	0.173	0.592	0.156	0.218	0.164
		30	0.593	0.205	0.238	0.215	0.576	0.121	0.196	0.140
	2	10	0.575	0.063	0.174	0.090	0.566	0.100	0.149	0.113
		20	0.576	0.060	0.191	0.088	0.592	0.107	0.233	0.128
		30	0.601	0.178	0.186	0.167	0.594	0.186	0.243	0.200
	3	10	0.584	0.066	0.208	0.097	0.558	0.064	0.132	0.083
		20	0.576	0.060	0.191	0.085	0.602	0.146	0.249	0.176
		30	0.609	0.206	0.236	0.196	0.601	0.212	0.233	0.213
HV	1	10	0.601	0.050	0.250	0.082	0.592	0.046	0.233	0.077
		20	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
		30	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
	2	10	0.675	0.238	0.283	0.244	0.642	0.117	0.221	0.147
		20	0.668	0.150	0.215	0.170	0.667	0.118	0.236	0.153
		30	0.628	0.064	0.250	0.099	0.628	0.066	0.250	0.100
	3	10	0.670	0.142	0.235	0.167	0.686	0.174	0.276	0.202
		20	0.670	0.093	0.220	0.128	**0.704**	0.149	0.275	0.182
		30	0.628	0.064	0.250	0.099	0.620	0.064	0.233	0.097

Table C.6: Classification performance measures for the CE-FCS approach with “all” pruning option using $\sigma_m = 0.1$ and $\lambda = 164e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.601	0.055	0.250	0.089	0.601	0.051	0.250	0.084
		20	0.618	0.178	0.263	0.191	0.618	0.177	0.263	0.190
		30	0.601	0.170	0.221	0.173	0.610	0.173	0.246	0.183
	2	10	0.601	0.053	0.250	0.086	0.601	0.050	0.250	0.082
		20	0.601	0.059	0.250	0.094	0.609	0.103	0.266	0.133
		30	0.610	0.131	0.245	0.156	0.610	0.101	0.245	0.137
	3	10	0.601	0.111	0.213	0.133	0.592	0.104	0.188	0.111
		20	0.584	0.107	0.179	0.123	0.592	0.084	0.189	0.109
		30	0.583	0.125	0.164	0.137	0.583	0.101	0.171	0.116
STD	1	10	0.576	0.179	0.124	0.133	0.592	0.206	0.173	0.158
		20	0.594	0.198	0.146	0.167	0.584	0.155	0.133	0.141
		30	0.583	0.147	0.121	0.132	0.626	0.237	0.233	0.225
	2	10	0.567	0.081	0.151	0.100	0.603	0.243	0.254	0.235
		20	0.567	0.110	0.151	0.103	0.593	0.132	0.201	0.144
		30	0.559	0.086	0.134	0.088	0.585	0.116	0.184	0.128
	3	10	0.617	0.209	0.238	0.216	0.585	0.202	0.183	0.176
		20	0.558	0.098	0.119	0.102	0.558	0.089	0.134	0.100
		30	0.576	0.137	0.144	0.137	0.559	0.092	0.119	0.101
HV	1	10	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
		20	0.601	0.050	0.250	0.082	0.601	0.050	0.250	0.082
		30	0.601	0.050	0.250	0.082	0.609	0.101	0.260	0.128
	2	10	0.670	0.131	0.260	0.167	0.592	0.081	0.213	0.110
		20	0.679	0.144	0.255	0.171	0.661	0.173	0.271	0.198
		30	0.662	0.131	0.248	0.157	**0.712**	0.248	0.345	0.281
	3	10	0.601	0.068	0.220	0.101	0.618	0.120	0.246	0.150
		20	0.687	0.193	0.256	0.215	0.626	0.100	0.208	0.130
		30	0.695	0.175	0.251	0.200	0.652	0.116	0.216	0.149

Table C.7: Classification performance measures for the CE-FCS approach with "max" pruning option using $\sigma_m = 0.1$ and $\lambda = 164e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.618	0.178	0.263	0.191	0.618	0.176	0.263	0.189
		20	0.592	0.146	0.211	0.151	0.592	0.145	0.211	0.151
		30	0.592	0.124	0.211	0.137	0.592	0.145	0.211	0.151
	2	10	0.601	0.058	0.250	0.093	0.601	0.053	0.250	0.086
		20	0.635	0.283	0.281	0.263	0.627	0.229	0.265	0.231
		30	0.626	0.231	0.250	0.228	0.626	0.233	0.265	0.234
	3	10	0.592	0.120	0.189	0.134	0.609	0.149	0.216	0.156
		20	0.575	0.116	0.139	0.117	0.575	0.098	0.146	0.111
		30	0.575	0.140	0.124	0.123	0.602	0.147	0.191	0.157
STD	1	10	0.602	0.213	0.178	0.185	0.609	0.151	0.221	0.173
		20	0.566	0.089	0.125	0.096	0.575	0.157	0.178	0.140
		30	0.575	0.121	0.133	0.120	0.594	0.216	0.228	0.207
	2	10	0.559	0.054	0.149	0.074	0.618	0.208	0.254	0.221
		20	0.576	0.135	0.169	0.140	0.585	0.115	0.201	0.129
		30	0.601	0.157	0.226	0.162	0.602	0.153	0.234	0.167
	3	10	0.550	0.089	0.094	0.087	0.601	0.172	0.246	0.192
		20	0.559	0.111	0.111	0.106	0.567	0.159	0.176	0.166
		30	0.592	0.125	0.186	0.142	0.575	0.201	0.144	0.147
HV	1	10	0.626	0.062	0.250	0.098	0.617	0.076	0.270	0.114
		20	0.617	0.102	0.270	0.133	0.626	0.083	0.279	0.123
		30	0.626	0.092	0.279	0.130	0.626	0.087	0.273	0.125
	2	10	0.712	0.208	0.279	0.231	0.650	0.180	0.283	0.211
		20	0.687	0.192	0.263	0.210	0.712	0.241	0.301	0.261
		30	0.695	0.244	0.288	0.253	**0.720**	0.253	0.311	0.272
	3	10	0.695	0.211	0.281	0.236	0.660	0.132	0.248	0.171
		20	0.679	0.164	0.260	0.191	0.679	0.172	0.275	0.200
		30	0.687	0.161	0.270	0.192	0.687	0.147	0.255	0.176

Table C.8: Classification performance measures for the CE-FCS approach with "all" pruning option using $\sigma_m = 0.1$ and $\lambda = 238e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.601	0.135	0.234	0.142	0.584	0.056	0.201	0.082
		20	0.609	0.158	0.244	0.161	0.609	0.167	0.244	0.162
		30	0.626	0.181	0.271	0.187	0.626	0.197	0.271	0.192
	2	10	0.575	0.172	0.146	0.154	0.602	0.231	0.176	0.195
		20	0.618	0.241	0.194	0.211	0.643	0.320	0.279	0.292
		30	0.626	0.245	0.204	0.219	0.643	0.285	0.264	0.269
	3	10	0.575	0.141	0.268	0.168	0.583	0.165	0.278	0.191
		20	0.566	0.137	0.211	0.157	0.558	0.109	0.201	0.122
		30	0.574	0.150	0.188	0.156	0.558	0.096	0.161	0.108
STD	1	10	0.617	0.287	0.269	0.269	0.617	0.267	0.251	0.248
		20	0.576	0.200	0.171	0.177	0.567	0.087	0.193	0.119
		30	0.583	0.178	0.194	0.174	0.574	0.126	0.201	0.143
	2	10	0.610	0.225	0.258	0.232	0.626	0.229	0.286	0.251
		20	0.627	0.216	0.278	0.240	0.644	0.304	0.368	0.327
		30	0.643	0.261	0.286	0.266	0.669	0.270	0.309	0.283
	3	10	0.584	0.189	0.269	0.208	0.601	0.210	0.289	0.221
		20	0.541	0.037	0.166	0.055	0.592	0.230	0.278	0.214
		30	0.574	0.218	0.213	0.203	0.584	0.178	0.216	0.177
HV	1	10	0.619	0.104	0.283	0.141	0.619	0.102	0.283	0.137
		20	0.609	0.108	0.260	0.136	0.609	0.099	0.253	0.129
		30	0.626	0.181	0.279	0.202	0.626	0.129	0.279	0.161
	2	10	0.669	0.180	0.269	0.211	0.670	0.178	0.269	0.208
		20	0.637	0.107	0.254	0.148	0.661	0.208	0.236	0.213
		30	0.636	0.145	0.239	0.172	**0.678**	0.191	0.264	0.219
	3	10	0.583	0.127	0.204	0.136	0.550	0.068	0.158	0.085
		20	0.567	0.137	0.211	0.150	0.533	0.036	0.101	0.042
		30	0.583	0.196	0.231	0.185	0.542	0.070	0.111	0.067

Table C.9: Classification performance measures for the CE-FCS approach with "max" pruning option using $\sigma_m = 0.1$ and $\lambda = 238e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.609	0.147	0.251	0.158	0.609	0.125	0.245	0.138
		20	0.617	0.150	0.246	0.159	0.617	0.107	0.255	0.140
		30	0.617	0.148	0.246	0.158	0.626	0.157	0.271	0.174
	2	10	0.651	0.323	0.256	0.284	0.660	0.300	0.266	0.279
		20	0.643	0.284	0.246	0.260	0.660	0.274	0.273	0.268
		30	0.675	0.313	0.278	0.291	0.667	0.298	0.268	0.274
	3	10	0.566	0.182	0.186	0.169	0.574	0.205	0.236	0.206
		20	0.591	0.202	0.221	0.208	0.583	0.256	0.204	0.207
		30	0.600	0.293	0.231	0.249	0.592	0.329	0.189	0.232
STD	1	10	0.576	0.154	0.148	0.146	0.576	0.117	0.194	0.145
		20	0.567	0.137	0.145	0.133	0.567	0.163	0.153	0.111
		30	0.542	0.079	0.101	0.074	0.533	0.042	0.076	0.041
	2	10	0.660	0.258	0.368	0.302	0.627	0.229	0.346	0.269
		20	0.617	0.202	0.239	0.210	0.618	0.210	0.258	0.215
		30	0.584	0.200	0.148	0.164	0.610	0.243	0.241	0.220
	3	10	0.541	0.038	0.166	0.053	0.584	0.128	0.266	0.160
		20	0.583	0.242	0.196	0.190	0.576	0.176	0.179	0.161
		30	0.583	0.195	0.218	0.173	0.576	0.134	0.219	0.143
HV	1	10	0.601	0.120	0.236	0.142	0.609	0.102	0.260	0.126
		20	0.651	0.191	0.283	0.215	0.626	0.137	0.259	0.165
		30	0.642	0.129	0.228	0.157	0.659	0.155	0.278	0.192
	2	10	0.635	0.202	0.206	0.201	0.643	0.139	0.205	0.164
		20	0.660	0.244	0.316	0.270	0.660	0.161	0.223	0.184
		30	0.652	0.235	0.266	0.248	**0.695**	0.250	0.296	0.269
	3	10	0.600	0.253	0.313	0.261	0.541	0.066	0.111	0.071
		20	0.566	0.123	0.181	0.131	0.583	0.153	0.231	0.163
		30	0.567	0.111	0.175	0.117	0.549	0.067	0.139	0.078

Table C.10: Classification performance measures for the CE-FCS approach with "all" pruning option using $\sigma_m = 1$ and $\lambda = 238e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.601	0.135	0.234	0.142	0.584	0.056	0.201	0.082
		20	0.609	0.158	0.244	0.161	0.609	0.167	0.244	0.162
		30	0.626	0.181	0.271	0.187	0.626	0.197	0.271	0.192
	2	10	0.575	0.172	0.146	0.154	0.602	0.231	0.176	0.195
		20	0.618	0.241	0.194	0.211	0.643	0.320	0.279	0.292
		30	0.626	0.245	0.204	0.219	0.643	0.285	0.264	0.269
	3	10	0.575	0.141	0.268	0.168	0.583	0.165	0.278	0.191
		20	0.566	0.137	0.211	0.157	0.558	0.109	0.201	0.122
		30	0.574	0.150	0.188	0.156	0.558	0.096	0.161	0.108
STD	1	10	0.617	0.287	0.269	0.269	0.617	0.267	0.251	0.248
		20	0.576	0.200	0.171	0.177	0.567	0.087	0.193	0.119
		30	0.583	0.178	0.194	0.174	0.574	0.126	0.201	0.143
	2	10	0.610	0.225	0.258	0.232	0.626	0.229	0.286	0.251
		20	0.627	0.216	0.278	0.240	0.644	0.304	0.368	0.327
		30	0.643	0.261	0.286	0.266	0.669	0.270	0.309	0.283
	3	10	0.584	0.189	0.269	0.208	0.601	0.210	0.289	0.221
		20	0.541	0.037	0.166	0.055	0.592	0.230	0.278	0.214
		30	0.574	0.218	0.213	0.203	0.584	0.178	0.216	0.177
HV	1	10	0.619	0.104	0.283	0.141	0.619	0.102	0.283	0.137
		20	0.609	0.108	0.260	0.136	0.609	0.099	0.253	0.129
		30	0.626	0.181	0.279	0.202	0.626	0.129	0.279	0.161
	2	10	0.669	0.180	0.269	0.211	0.670	0.178	0.269	0.208
		20	0.637	0.107	0.254	0.148	0.661	0.208	0.236	0.213
		30	0.636	0.145	0.239	0.172	**0.678**	0.191	0.264	0.219
	3	10	0.583	0.127	0.204	0.136	0.550	0.068	0.158	0.085
		20	0.567	0.137	0.211	0.150	0.533	0.036	0.101	0.042
		30	0.583	0.196	0.231	0.185	0.542	0.070	0.111	0.067

Table C.11: Classification performance measures for the CE-FCS approach with “max” pruning option using $\sigma_m = 1$ and $\lambda = 238e5$

CRM	ζ	k	k_{nnc}= 1				k_{nnc}= 3			
			Acc	Prec	Rec	F-s	Acc	Prec	Rec	F-s
SD	1	10	0.609	0.147	0.251	0.158	0.609	0.125	0.245	0.138
		20	0.617	0.150	0.246	0.159	0.617	0.107	0.255	0.140
		30	0.617	0.148	0.246	0.158	0.626	0.157	0.271	0.174
	2	10	0.651	0.323	0.256	0.284	0.660	0.300	0.266	0.279
		20	0.643	0.284	0.246	0.260	0.660	0.274	0.273	0.268
		30	0.675	0.313	0.278	0.291	0.667	0.298	0.268	0.274
	3	10	0.566	0.182	0.186	0.169	0.574	0.205	0.236	0.206
		20	0.591	0.202	0.221	0.208	0.583	0.256	0.204	0.207
		30	0.600	0.293	0.231	0.249	0.592	0.329	0.189	0.232
STD	1	10	0.576	0.154	0.148	0.146	0.576	0.117	0.194	0.145
		20	0.567	0.137	0.145	0.133	0.567	0.163	0.153	0.111
		30	0.542	0.079	0.101	0.074	0.533	0.042	0.076	0.041
	2	10	0.660	0.258	0.368	0.302	0.627	0.229	0.346	0.269
		20	0.617	0.202	0.239	0.210	0.618	0.210	0.258	0.215
		30	0.584	0.200	0.148	0.164	0.610	0.243	0.241	0.220
	3	10	0.541	0.038	0.166	0.053	0.584	0.128	0.266	0.160
		20	0.583	0.242	0.196	0.190	0.576	0.176	0.179	0.161
		30	0.583	0.195	0.218	0.173	0.576	0.134	0.219	0.143
HV	1	10	0.601	0.120	0.236	0.142	0.609	0.102	0.260	0.126
		20	0.651	0.191	0.283	0.215	0.626	0.137	0.259	0.165
		30	0.642	0.129	0.228	0.157	0.659	0.155	0.278	0.192
	2	10	0.635	0.202	0.206	0.201	0.643	0.139	0.205	0.164
		20	0.660	0.244	0.316	0.270	0.660	0.161	0.223	0.184
		30	0.652	0.235	0.266	0.248	**0.695**	0.250	0.296	0.269
	3	10	0.600	0.253	0.313	0.261	0.541	0.066	0.111	0.071
		20	0.566	0.123	0.181	0.131	0.583	0.153	0.231	0.163
		30	0.567	0.111	0.175	0.117	0.549	0.067	0.139	0.078