Behaviour Recognition and Monitoring of The Elderly using Wearable Wireless Sensors

Dynamic behaviour modelling and nonlinear classification methods and implementation

Jonathan James WINKLEY

PhD

School of Computing, Informatics and Media

University of Bradford

2013

Abstract

BEHAVIOUR RECOGNITION AND MONITORING OF THE ELDERLY USING WEARABLE WIRELESS SENSORS

Dynamic behaviour modelling and nonlinear classification methods and implementation

Keywords
Ambient Assisted Living, dimension reduction, classification, state probability.

In partnership with iMonSys – an emerging company in the passive care field – a new system, “Verity”, is being developed to fulfil the role of a passive behaviour monitoring and alert detection device, providing an unobtrusive level of care and assessing an individual’s changing behaviour and health status whilst still allowing for independence of its elderly user. In this research, a Hidden Markov Model incorporating Fuzzy Logic-based sensor fusion is created for the behaviour detection within Verity, with a method of Fuzzy-Rule induction designed for the system’s adaptation to a user during operation. A dimension reduction and classification scheme utilising Curvilinear Distance Analysis is further developed to deal with the recognition task presented by increasingly nonlinear and high dimension sensor readings, and anomaly detection methods situated within the Hidden Markov Model provide possible solutions to identification of health concerns arising from independent living. Real-time implementation is proposed through development of an Instance Based Learning approach in combination with a Bloom Filter, speeding up the classification operation and reducing the storage requirements for the considerable amount of observation data obtained during operation. Finally, evaluation of all algorithms is completed using a simulation of the Verity system with which the behaviour monitoring task is to be achieved.

Acknowledgements

I would like to thank everyone at iMonSys for their time, creative ideas and insightful input in the Verity project in which this research finds an application.

Special and extensive thanks to my supervisor, Ping Jiang, whose support has been invaluable during my entire time at the University of Bradford and without whom this work would not have come to fruition; thanks also to Alamgir Hossain for his input in the initial stages of the research, to Rona Wilson for her continued administrative support and to all examiners for their time.

I would finally also like to thank family and friends for their support and encouragement throughout, and especially thank and dedicate this work to a select, close few who provided me with many welcome distractions and much-needed reassurance that I could do it.

Contents

List of Tables

List of Figures

Chapter 1
Introduction

1.1 Background and Motivations

The number of elderly and infirm living in sheltered accommodation is increasing, with more people of retirement age in the UK choosing to "age in place" with some form of support - 473,000 in 2008/2009 [1] - yet in figures calculated by Help the Aged, the number of those actually being supported decreased by a dramatic 13% in the years 2000 to 2006 [2] with the trend declared likely to continue in successive years. This has been attributed to local authorities being given the right to restrict access to home help if those requesting it did not meet their imposed "eligibility criteria" [3], essentially resulting in those assessed as "critical" cases receiving financial assistance for home care and the "low" or "moderate" receiving little to no help. At the same time, AgeUK [2] noted that "17% of older people have less than weekly contact with family, friends and neighbours". These facts and figures show that there is increased risk for those not being monitored or personally cared for: from minor incidents in the home, from illness which causes immobility or from other unforeseeable scenarios which as such would go undetected if no contact is made with the individual over a long period of time.

Many monitoring systems currently exist for installation in the homes of the elderly and infirm who require monitoring but cannot afford direct care from a "home helper" or the fees required for care in a residential home [4]. The need for such a monitoring system can be suggested by either the individual wanting help, or in many cases the concerned family of such an individual. Systems range from simple, wearable pendant and button devices worn around the neck or wrist for emergency use [5], whereby if the user experiences difficulties, at the press of the button the home-inbuilt system can call a nominated contact who can then communicate with the individual; to the intelligent systems which monitor the entire home through use of smart sensing and visual monitoring, using information from distributed sensors to formulate a status estimation of the user [6 , 7]. These “Ambient Assisted Living” (AAL) systems all have underlying algorithmic and computational frameworks and principles on which they rely, either simple non-intelligent means where personal monitoring is triggered by the user (i.e. the pendant systems) or more intelligent methods using numerous mathematical techniques to reach logical estimations of the current status of the user, as exemplified by the smart home-integrated systems above. The intelligent methods are researched and documented extensively in the literature in the healthcare, biomedical engineering and applied mathematics fields [8]; each technique is suited specifically to different applications requiring a form of behaviour or sequence monitoring with a determinable “state” output. A significant limitation of the current passive behaviour monitoring systems developed for the elderly population however, is in the lack of consideration of human reasoning for the determining of states from observable data even with the intelligent methods documented in the respective fields; the majority (as evident even from a short literature review) utilise high dimension training examples with which to infer a state belief based purely on correlations in data, where in fact such correlations may be indicative of some other state when viewed contextually within a sequence [9], or the high dimensionality obstructs the true determinable state [10]. Human reasoning is still a considerably effective employable tool for these uncertain situations in state determining, when the subject being observed is human and has the ability to provide an indication themselves of their own current status – an overlooked aspect of human behaviour monitoring that this work intends to utilise and build upon. The issue of high dimensionality obstructing pattern recognition is approached and overcome here with a dimension reduction and classification scheme enabling linear separation of observation data in order to further support the reasoning provided by a user in an alert scenario.

A key requirement of intelligent passive monitoring systems is that they not only provide the estimation of a state based on the pure observable data, but are able to identify states which do not fit the determined “normal” behaviour of the user so as to alert the relevant service or care provider of an instance where some form assistance is necessary. Within these devices therefore can typically be found numerous methods employed for the action of such tasks: static and dynamic behaviour probability models, anomaly and outlier detection models and fault diagnosis methods. The software system developed for this work incorporates variations of such models and methods, combining them with an additional human reasoning ability through intelligent programming and direct voice communication with the user – an approach to passive behaviour monitoring not previously considered and documented in the literature of the area or products available on the commercial market.

Combining the hardware developed alongside the research with the intelligent methods developed herein, we produce Verity: a passive system to be used alongside traditional methods of monitoring elderly persons who choose to live without 24 hour personal care (i.e. home visits, medical health checks, one-to-one telecommunication), yet still require a level of round-the-clock observation for assurance of their wellbeing through Ambient Assisted Living. The hardware consists of a wearable sensor system and a mobile phone specially developed for the application, affording the user the ability for behaviour monitoring external to the home unlike with previous devices requiring home integration; the software ensures a definitive decision regarding the user's state through execution of newly developed and combined dimension reduction, classification and optimisation techniques overcoming the uncertainty previously encountered with other systems. Verity indeed requires extensive testing of each built-in function with the developed software before being acceptable for use in a trial - where key data can be obtained to further aid in its on-going development past the limitations of this research.

1.2 Aims

The overall aim of this research is to investigate, develop and optimise suitable intelligent recognition methods to be incorporated into a specialised behaviour monitoring device, such that the information made available to it through the ambient sensing of a user is adequately utilised to inform a belief of their current behavioural state. The proposed scheme is required to combine multiple sensor readings and ascertain a likelihood of emission from each behavioural state such that the most likely at any time point within the monitoring window can be determined according to the observations of the device. The current Hidden Markov Model is modified to account for the multidimensional, nonlinear and significantly dynamic observations from the often-uncertain readings provided by the sensors in order to inform of the state belief, by implementing a method of fusion which ensures an element of human reasoning inherently informs of each decision (Chapters 4 & 5).

The modifications will increase the usability by a standard user however they will also increase computation and storage space in the short term so the secondary aim focusses on the optimisation of the first such that the result is usable not only in the behaviour monitoring scheme but in other similar fields requiring similarly applied methods of state determining. The developed solution is then capable of overlooking the linearity (or lack thereof) of the training data and learning through pure exposure to the data set whether or not a new observation is classifiable to one of the already learned human-reasoned classes, with a significant speedup in real-time performance over previously developed methods documented both in the literature and as part of this research (Chapters 6 & 7).

1.3 Contributions

I. A combined Hidden Markov Model and Fuzzy Inference System is developed as a means of incorporating a higher element of human reasoning in the behaviour monitoring device, in which an error detection scheme is further created and integrated to adequately identify instances where the determined state is not as expected (Chapters 4 & 7).

II. A new method for pattern recognition of nonlinear and high dimension sensor data is proposed through the application of a dimension reduction and classification technique to the original raw data, with details of its implementation and results of its experimental operation included to establish its effectiveness when compared to standard Neural Network techniques of classification of such data (Chapter 5).

III. A new method of Instance-Based classification based on established means is developed to address the problem of computation speed and storage requirements when large data sets are involved, with results documented to support the claims of superiority in situations previously utilising other such methods (Chapter 6).

1.4 Thesis Outline

Chapter 2 contains the literature review of applications and systems relevant to the Ambient Assisted Living device developed in this research. Background information and implementations of the mathematical techniques utilised and built-on herein is also provided as a means of introducing and explaining their relevance with respect to the application.

Chapter 3 documents and explains the hardware of the system along with the preliminary control methods and algorithms required to provide subsequent state-determining operations with the necessary information.

Chapter 4 details the Hidden Markov Model and Fuzzy Inference system developed in first stage of the research which facilitates the state-determining based on the sensors' observations of the user and how the linguistic "rules" that govern how the combined observations are dealt with interface with the established model.

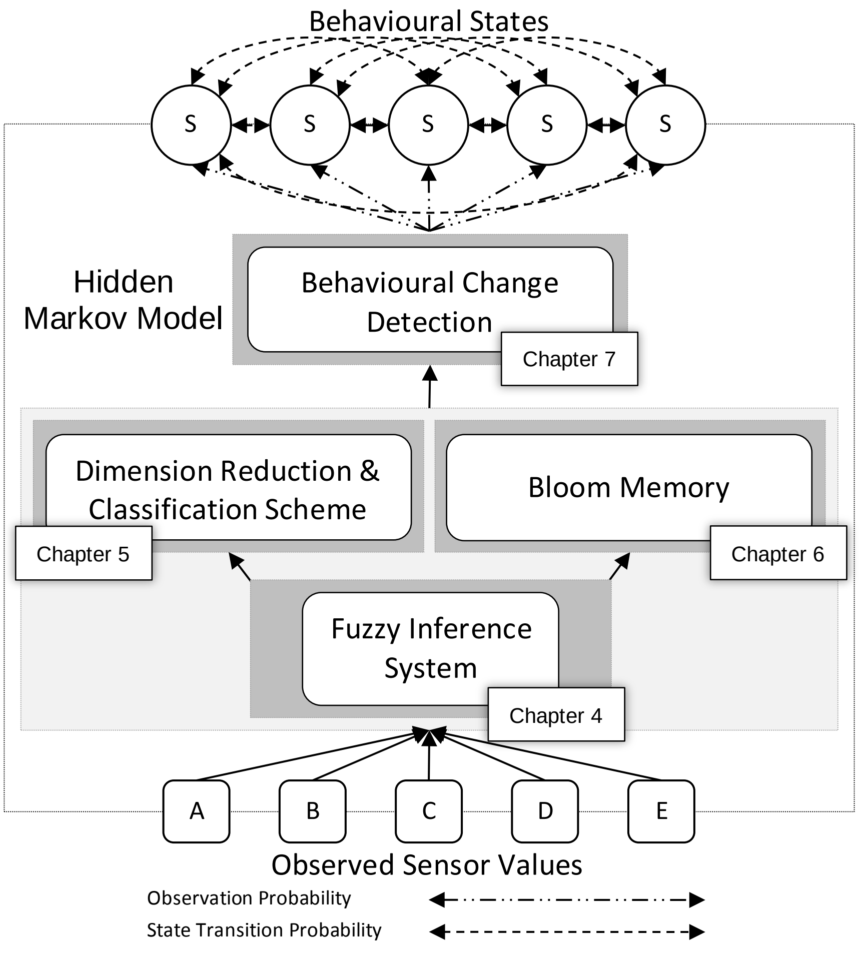
Chapter 5 deals with the creation of the dimension reduction scheme used to facilitate the classification operation of the raw sensor data in a high dimension space in which it has no linear manifold, which in turn helps both visualise the input data that informs of a state belief and enables the model to update autonomously; thus removing the need for employing the linguistic rules which govern the fusion of the initial sensor observations by reducing the probability operation to a simple mathematical decision based on trained data.

Chapter 6 optimises the classification operation for high dimension data resulting in a compressed format in which the training data is stored and subsequent unseen values are classified according to previous sightings within the training set. The method allows for a more streamlined system which is capable of expanding the storage of data as the system is used and is influenced by the memory techniques employed in the human brain.

Chapter 7 provides a behavioural change detection scheme for the combined Hidden Markov Model-based system: utilising the model's output as expected normal behaviour the method is capable of detecting outliers within the observations and state decisions in order to raise alert scenarios. Example scenarios are given where these methods can detect possible fault situations with a user and the combined system incorporating all aspects of the research is tested in simulation using observations obtained with the hardware in order to provide evidence of its potential suitability in a real scenario.

Finally, in Chapter 8, concluding remarks are made with respect to the overall implementation of the combined system before limitations are identified and further work suggested based on the techniques developed in the preceding pages.

The framework of the system itself utilising the Hidden Markov Model, its constituent components and their location within this thesis is provided below.



Chapter 2
Literature Review

This section of the thesis reviews related applications and solutions of relevance to the application and problems researched in this work.

First, current projects in the Ambient Assisted Living field of research are reviewed as a means of providing a background to the Verity project, before the Hidden Markov Model - the primary algorithmic mechanism within this application - is considered and explained, along with a significant overview of applications in the same field which also utilise a similar methodology in the determining of a state probability. Subsequently, similar applications which employ a different control scheme are also reviewed, and distinctions made as to why the Hidden Markov Model method is more suited to this system. Following logically from this starting point, the use of a multi-sensor system for behaviour monitoring is considered literarily, before the issue of data dimensionality is considered as a precursor to the classification, optimisation and learning schemes that are then detailed in relation to that developed as part of this research. Finally the subject of errors is addressed when dealing with the identification of an anomalous value within a series, in both the behaviour monitoring and manufacturing fields that inspired the development of the schemes for this research.

2.1 Recent Ambient Assisted Living Projects

In recent years, the increased requirement for elderly behaviour monitoring through technological solutions has generated an entire section of academic research devoted to the development of Ambient Assisted Living (AAL) systems, as evidenced by this literature review. AAL projects typically have considerable academic involvement and as such can be found spanning multiple institutions and have strong partnerships with the commercial sector; but whilst their research is all but guaranteed to apply to a specific device or system, the methods and practises implemented in the behaviour monitoring task are transferrable between systems and help inform the development of successive devices and on-going research into the AAL field.

The Independent LifeStyle Assistant™ (ILSA) was a Honeywell Laboratories research project into AAL intended to develop a system enabling monitoring and support of the elderly population wishing to live longer and independently in their homes, as opposed to being institutionalised as a means of receiving round-the-clock care and assistance [11]. The system was conceived to be wholly relied upon by the elderly user: providing monitoring of the home environment, health status, fall detection, medication reminders and even assisting in the activation of lighting within the home. The project avoided the detailed analysis of a user with respect to the medical conclusions reachable by observation of vital signs etc. through ambient sensing, instead focussing on the similar assistance provided by an in-home caregiver – alerting the required authorities if safety is compromised and allowing off-site caregivers to access information regarding the user and their wellbeing in their own home.

The multi-agent system incorporated only passive devices, with occupancy and location determining being accomplished through use of motion sensors installed within the experimental environment. The user interface was considered to be pioneering at the time (the project ran 2000 – 2003), having consisted solely of a portable touch-screen with wireless access to the internet as a means of interacting with and reminding the elderly user of necessary actions in much the same way as a physical caregiver would. Initially - as with the Verity device of this research – an additional mode of interaction was considered to be through telephone communication, however testing and trials returned results to indicate that in this in-home system, the visual feedback from the touch-screen was more effective. This aspect of the research highlighted that the method in which a message is delivered and the voice used to communicate it requires careful consideration so as not to alienate or unnerve the elderly user – automated telephone interactions were identified to be "cognitively overwhelming" for the elderly user [11]. Verity's decision and verbal interaction trees however are designed to be amiable and comforting for the user, and intend to overcome the barriers possibly experienced with the ILSA project due to the increased exposure of elderly people to such voice-activated systems utilised by modern technology not prevalent at the time of Honeywell's study.

The Information Technology for Assisted Living at Home (ITALH) project was a collaboration between Tampere University of Technology, Finland, University of Berkely, California and Aarhus University, Denmark that incorporated the sub-project, UUTE: utilising novel sensors, wireless communication and service platforms to provide a level of support and monitoring for selected groups of patients requiring assistance in daily living [12].

The UUTE project developed a home network consisting of specialised sensor nodes and a "home client", connected to a wireless network which was in turn linked to an external storage server. Nodes were home-integrated and user-triggered, allowing for the collection of data from a weight scale, blood pressure and ECG data and a bed sensor capable of monitoring sleeping activity provided information regarding heart rate and respiration. Communication between nodes and the end user was based partially on the Zigbee wireless protocol, with information obtained from each sensor processed by the home client and sent to storage on its local hard drive. The data was then transmitted on a daily schedule to a UUTE server, where it is viewable by caregivers and those capable of providing support. With intentions of being able to detect motion, falls, health status and provide a level of motivation for possible "rehabilitation" of users, the UUTE system perhaps differs from Verity only in its increased reliability on home integration of sensors and nodes.

Despite the project therefore primarily being an exercise in Wireless Sensor Network (WSN) integration within the home as a means of building a platform for the monitoring application, the conclusions and experiences obtained help inform of future possibilities and limitations of such a system; as considered with Verity, wearable sensors communicating with integrated nodes wirelessly in the UUTE system presented issues with power consumption and data quality. For this reason the communication protocols incorporated within Verity were specially selected and the hardware sourced for the purpose of wearability and longevity of power – enabling a more independent living style for the user without their need to concern themselves with being monitored only when in the house or near a power supply.

Another significant AAL project is that of the Service Oriented Programmable Smart Environments for Older Europeans, SOPRANO: a consortium consisting of 24 partners from 7 countries, including small businesses, public companies and academic institutions all working towards researching and producing devices and systems capable of improving the quality of life for the elderly in Europe. The consortium structure and skillset enables SOPRANO to access state-of-the-art technologies in order to produce a platform capable of providing a flexible and personalisable solution to elderly monitoring [13].

Within the SOPRANO project, a smart home which incorporated sensor nodes and actuators to assist in the care and independent lifestyle of the elderly user was proposed: user-identified issues informed of the platforms' need for incorporation of methods of fall detection, household equipment control and user interaction to ensure safety and security [14]. This approach to AAL is very user-involved and does somewhat encroach on the idea of a passive monitoring system; with the control of the environment forming a key aspect of the device, it poses a moral question as to whether or not it is indeed assisting the elderly with an independent lifestyle. Verity maintains the independence of the user by simply co-existing with them whilst ensuring a level of monitoring occurs – thus ensuring passive care and assistance when required through alert triggering to caregivers.

The PERSONA (Perceptive Spaces Promoting Independent Aging) project and consortium dealt with the design of self-organizing middleware for an AAL platform intended to be fully scalable over its period of use to account for a multitude of monitoring scenarios [15]. Daily activities were monitored and observed using sensors located in and around the user's environment, allowing for interactions with household objects, room occupancy, posture identification and location to be detected and form a status estimation for the user [16]. Many technologies were incorporated into the PERSONA systems [17], with the well-established communication protocols of 3G, Bluetooth and WiFi forming the basis for the WSN; where emergent technologies were identified as desirable for inclusion within an AAL system – textile sensors, Zigbee communication and Speech Recognition were all highlighted as significant enhancements for behaviour monitoring of the elderly. In particular we find the latter technology of speech recognition to be most applicable to Verity: given the already identified need to communicate with an elderly user in the most effective way possible, speech recognition was identified in the PERSONA project as a method of providing the end user with the ability to adapt the monitoring system to their own specific needs [15].

Developing on the notion of speech recognition for the task of elderly monitoring, the Voice Controlled Assistive Care and Communication Services for the Home (vAssist) project was initiated in 2011 with the goal of providing voice controlled home care and communication services for elderly suffering from chronic diseases and those with motor skill impairments. The intention is to develop interfaces and communication applications using natural voice interaction, with the aim of reducing the cost of existing hardware used in elderly care and enhancing the quality afforded to them with current technology [18].

The AAL Forum has also begun the ASSAM (Assistants for Safe Mobility) project [19], which builds on the notion of mobile devices assisting elderly users and those with declining cognitive capabilities. The assistive systems will range from wheelchairs to handled-walkers embedded with smart interfaces to enable multi-sensor observations and data to inform a user of their location and notify of impending dangers etc. through use of environmental scanning and interaction with devices around the home. With the natural language interaction, this project further combines the principles outlined in the aforementioned others in order to provide a user with a more convenient and amiable approach to assistance whilst maintaining a standard of independent living. Despite not performing much in the line of behaviour monitoring, the passive devices and integration primarily with the user rather than the environment are similar in execution to those of the wearable Verity device – existing solely to enhance the user experience without becoming intrusive.

With the MobileSage Project [20], also an AAL consortium development, we find a truly mobile and hand-held system with functionality although significantly dissimilar to Verity, having a form factor evidently identified as worthwhile to employ for usage by the elder generation. The case usage scenario in [21] highlights an elderly user who whilst not afraid of technology, has no wish to learn a new skill set in order to use a device or system that may assist her everyday life. The concept is a “Help-on-Demand” application embedded within a smartphone, and with access to a Content Management System which stores the user profile and details regarding the objects or technological devices for which the user requires help. The system delivers personalised instructions via the smartphone, possibly utilising natural language synthesis and motion graphics, as a means of assisting the user through the process of interaction with the everyday object which is identified using Near Field Communication (NFC) tags located in that area. The mode of delivery and interaction with the user is akin to the Verity system and serves as a means of making life easier for the elderly who find daily tasks difficult perhaps due to eyesight loss, or motor skill impairments. Verity has the ability to also provide such users with assistive information such as location, or provide feedback regarding observable vital signs regarding their health. However, despite the slew of projects in the AAL field, Verity sits comfortably in an area not currently occupied by a device or system capable of fulfilling the mobile, discreet, passive behaviour monitoring task required by elderly users.

2.2 The Hidden Markov Model for Behaviour Monitoring

The above identified AAL devices and systems each utilise different intelligent frameworks with which to execute the tasks for which they are intended, be it assisting in daily life through increasing mobility and awareness of surroundings (ASSAM, SOPRANO and MobileSage) or through the passive behaviour monitoring employed to provide caregivers with alerts regarding their elderly users' health (vAssist, PERSONA, UUTE).

Specialised sensors are utilised in such systems to provide the most accurate numerical representation of a specific, observable property of the user – be it through observation in the home using distributed sensors or as part of a wearable network, with each sensor performing a different task to inform of a user's behavioural state. The passive behaviour monitoring is here the focus of the research for the Verity project, developing a system capable of observing the user through the wearable sensors as a means of determining current behaviour and wellbeing status.

As the behaviour of the user is not directly observable through sensing, it is therefore hidden and can only be *estimated* based on collaborative data from sensor observations. For example, a typical wearable motion sensor would be in the form of an accelerometer which, whilst able to indicate that movement is occurring, is not able to state definitely that the motion being observed is specifically walking or running or is a symptom of some motor-related disability [22]. This problem is tackled in behaviour monitoring applications through use of multiple sensors and intelligent, programmable estimations of state based on probabilities of occurrence or through preconceived notions of behaviour by a trained professional [23 , 24 , 25].

What is required for the autonomous passive behaviour monitoring is knowledge of how a behavioural state is typified by a sensor's observation, before applying some form of estimation process to reach a conclusion as to what state the user is exhibiting based on the combination of each sensor reading. Human behaviours are continuously changing: in a real-time, around-the-clock monitoring system, the sensor observations and their combinations change constantly and the behaviour monitoring problem becomes a dynamic one, resulting in the necessity for a model capable of handling such a case.

The Hidden Markov Model (HMM) [26 , 27] is a statistical model, which takes as its input a visible observation emitted from a hidden state. The goal of an HMM is to determine this hidden state producing the observation, which is itself typically located in a sequence of states with the observations forming a sequence of model inputs. For reference, its parameters and implementation are detailed in Appendix A. The standard variation of a Markov Model, however, differs in its visibility of the emitting state: the goal of such an implementation is typically to predict a future state based on current state alone, with the past states being viewed as independent and having no relationship to the decision of future possibilities.

With the ability to recover a hidden data sequence from only the visible observations resulting from the data itself, the HMM is utilised in a broad spectrum of applications not necessarily concerned with monitoring. Within the bioscience field, for example, the model is ideal for gene prediction - where each state emits random DNA strings of random length, which are observable as a means to determine the gene producing them [28] - and in protein structure prediction and genetic mapping [29]. Cryptanalysis and cryptography benefit significantly from the utilisation of Hidden Markov Models [30], with hidden states representing internal states of countermeasures and outputs being the observations of the side channel [31]; and in the measurement of partial discharge (PD), the time-varying and sequential properties lend themselves to be modelled with an HMM such that PD patterns can be classified to inform of insulation system defects [32].

The Hidden Markov Model is therefore well suited to applications where sensors can obtain observations of the user, either directly through personal contact or indirectly through visual monitoring, for the purposes of determining their current state. The application of the models by [33] and [34] can be termed indirect, with information obtained from video streams and the still images which constitute them. A major aspect of both of works is concerned with the context in which the behavioural state is observed. The use of a hierarchical architecture HMM [35] allows layers to consider various aspects of the detected behaviour. The deduction is that for true detection and understanding of behaviour from the observation, the spatial and temporal contexts must be considered along with the activity itself.

The sequential nature of speech means that the HMM is ideal for applying to speech recognition. Rabiner [36 , 37] notes that the left-right HMM therefore has the desirable property that it is capable of modelling such signals where observable properties change over time. When the first element of a spoken word is observed, there can be given a distinct probability of hearing another sound in sequence. This is termed as left-right because the states viewed must progress logically from left to right to ideally model the sequence i.e. no state may transition backwards. The ergodic model however is usually applied to the behaviour recognition tasks of the monitoring systems reviewed here: realistically speaking, the initial state of the monitoring system is only important in the first few iterations of the model; as almost all states can allow transition from one to another the observation sequence can only logically belong to a limited (if not single) number of state sequences as the model progresses over time.

The initial and key concern with Verity is in the identification of the state/behaviour sequence exhibited by the elderly user of the system, where only the sensor observations and user-interaction are available to reach a logical estimate of their current status. This sequence further enables in the detection of problems and issues experienced on a daily basis, as the disparity between the prediction of a state and the actual observation of a state can indicate that the user is exhibiting an unusual behaviour not expected as part of their current trend of states.

The sequence problem is often considered in the literature in articles detailing an HMM application unrelated to behaviour monitoring. Bengio details many applications of HMMs to sequential data with particular interest taken in speech recognition [38]. Bengio considers the Viterbi algorithm [39] as the most suitable method for sequence determining with an HMM in data with a low number of state transition probabilities. This would indicate that for inclusion in a behaviour monitoring system where the number of distinctly dissimilar states can be found to be small, the Viterbi method is ideal. For speech recognition where the number of transition probabilities is remarkably higher given the complexity of the data being modelled, other graph search methods are seen to be more appropriately applied. Chung and Liu [33] successfully applied the Viterbi algorithm to the behaviour recognition task, with specific application to the activity extraction layer of their model.

Li [40] actually proposes a method that is said to outperform the traditional HMM and its sequence-determining in some cases, as more of the information available is used to reach a conclusion regarding the current state. The method is derived from applying probability theory to the original model, where two fundamental properties of HMMs are considered from which the variation of the HMM is recognised. Li notes the wide variety of applications of HMMs and their common characteristic of the determining of states based solely on the immediately preceding state. The proposed HMM with States Depending on Observations (HMMSDO) uses not only the preceding state but the preceding observation to formulate a state belief. It is notable in the paper that when the method is applied to the prediction of protein secondary structures, the HMM may achieve better results except in cases where training data is large; a property presumably explainable by the fact that the HMMSDO incorporates more parameters which only after numerous training iterations will have a noticeable improvement over the traditional model. The new model structure could well be applied in the behaviour monitoring of individuals where the volume of data available for training may be suitably large and could even be considered for the Verity system developed here once the initial stages are successfully implemented and the amount of data used for calculation of state sequences is deemed large enough to benefit from this further modification.

The deliberation over use of the HMM and hierarchical models for behaviour modelling can be attributed to the differences in the definition of human behaviour. Atallah and Yang conducted a survey of pervasive sensing applied to behaviour profiling [41], identifying that activities can not only be viewed as belonging to a consecutive sequence (as assumed by many machine learning techniques), but also as concurrent and interleaving. For example, with concurrent scenarios in behaviour monitoring terms being those where multiple complex activities occur at the same instance, a user standing up to cook in the kitchen whilst simultaneously talking to a person in the next room is viewable as 3 possible states existing in parallel. An interleaving state scenario would, building upon the above example case, include the instance where the user halts their cooking process momentarily to perhaps walk to the refrigerator to obtain some ingredients before returning to the cooking task and resuming the previous concurrent activities. This poses a challenge for machine learning techniques which only model the sequential properties of behaviours - assuming a probability of transitioning from one state to the next rather than incorporating assumptions of multiple behaviours either existing at one instance, or in a cyclical format where an activity is interrupted as a means to accomplish another. Within the Hidden Markov Model however there is an ability to view the above scenarios as singular state instances, with the observations regarding the user's motion and physical signs indicating a "kitchen usage" state, or "cooking" state which incorporate all above sub-states, thus overlooking the complexity of combining multiple states to deduce behaviour.

Subsequently, it is recognised in the survey that there is "across subject variability" which is attributed to the varying abilities of subjects both physically and mentally. Such differences cause models to react in differing ways but if the model is incapable of catering for the changes then the behaviour will be identified incorrectly. The pervasive characteristics of such monitoring devices are also considered, with respect to the level of security offered to the user by devices capable of viewing them directly and inferring status based on their posture and/or movement. The concern arises not only due to the method of storage of the data being utilised, but in the manner in which it is gathered. This problem is lessened in [42] even though the method uses vision to infer a belief. The image of the user obtained by a visual sensor is converted into a non-descript blob shape which extracts them from their environment. This allows sufficient information to be gathered regarding their location and posture, but also has the benefit of reducing the user's concerns of privacy given that the image cannot be reconstituted into a full and accurate representation of them.

2.3 Multi-Sensor Fusion

With wearable sensor systems employed for passive behaviour monitoring, observations are produced by each sensor concurrently and recognition of a state therefore involves estimation based on the multiple readings being taken as a single observational input.

The case of multiple sensor observations being used as single (vector) inputs to an HMM such as that identified for use in this research is not commonly addressed in the literature; instead, behaviours or states are usually inferred solely from a single observation, with a probability of occurrence for a state being defined explicitly at the time of model creation. Therefore, with the Verity implementation utilising multiple sensors, the fusion of the multiple sensor observations into a single, useful value is required in order to successfully exploit the HMM's ability to identify the most likely behaviour.

The traditional HMM uses probability distributions or discrete probability values assigned to single observations. In the behaviour recognition task, more detailed models take observations from a variety of sources to ascertain an intelligent estimate of the hidden state. When the hidden state can be determined with greater accuracy if a number of observation sources are reviewed, the fusion of such inputs must be considered [43 , 44 , 45]. What must be taken into consideration, however, is that this fusion of multiple sensors can in some cases produce worse results than the output of the best single sensor ("catastrophic fusion", [46]), due to the possibility of inaccurate sensor readings being combined with those evaluated to be more accurate [44]. Before use in the HMM, the numerous readings must be effectively fused to best reflect the information they convey when viewed together. The weighting scheme for the fusion of sensors must be carefully considered before their inclusion as an observation, so as not to over emphasise the significance of one sensor when compared to another otherwise the above stated problem may arise. Non-discriminant fusion is desired in applications where each sensor reading has an equal influence on the determining of the state; that is to say that the weight applied to each reading is calculated so that the fusion distribution is a pure summation of individual sensor readings.

Discriminant sensor fusion in multi-sensor systems has two modalities of operation [47]. Centralised fusion concerns the collection of sensor data from the individual nodes distributed in the system and combining the values at a central location, where the inference of a state is performed using the data available to this central point. Decentralised fusion operates without a central fusion site, with each sensor site (or multiple sites in a single system) incorporating its own method of fusion to determine how much value to contribute to the state inference process. Decentralised Fusion results in a scalable and modular system, with the ability to easily increase the number of nodes contributing to the behaviour determining task.

The Dempster-Shafer Theory (DST) of data combination incorporates a level of “ignorance” into the discriminant fusion process, with evidential information informing of the plausibility of an independent observation’s occurrence - whereby the resultant combined observation regarding a state is specified as a belief function rather than as a probability distribution. In the military-applied Target Identification application of [48], the authors compare the DST to the Bayesian method of sensor fusion, concluding through experimentation that the former’s evidence-based approach is slower to reach a state decision than the latter which utilises probabilities of occurrence in its fusion of multiple sensor values. It is in Bayesian Theory that we find the Hidden Markov Model and thus identify the most appropriate form of fusion for behaviour monitoring applications with multiple sensors to involve probabilities rather than evidential beliefs.

Broadly speaking, like many passive behaviour monitoring systems Verity consists of a central hub in which the processing of the multiple sensor readings occurs and behaviours are inferred; however whilst each sensor is independently capable of providing a useful reading, the collaboration of one or more sensors is required as a means of ensuring reliability in the combined fusion task. Discriminant fusion methods adjust this output observation based on multiple readings informing of the reliability of each sensor’s performance. In the case of wearable sensor systems, the motion of users and environmental factors can impact on a single sensor’s ability to provide a useful result. Contact, pressure, and temperature sensors can all be affected considerably by the motion of a user: disconnections between the sensing surface and the user can result in periods of null values, and high acceleration may ramp a sensed value to its maximum – both instances causing considerable issues in fusion circumstances when averaging operations are used to identify periods of activity.

Kalman Filtering is useful in the situation where one sensor's reliability may be called into question until compared with observations obtained from another, independent sensor through time series analysis of both sensors. The filter is implemented in many systems requiring precise information from imprecise observations, such as in GPS navigation of consumer vehicles [49], and in robotic applications [50], where the dynamic weighting of the sensor evidence which has an ongoing expectation value regarding positioning and location, for example, is used to estimate the current state of the system. The requirement for implementation is that all dynamics of the system and noise are known, with the estimation error being minimised when the noise causing the uncertainty is white (Gaussian) noise [51]. In actuality, the Kalman filter is somewhat akin to the Hidden Markov Model in the fact that what is being found is essentially a "hidden state" based upon observations with probabilities of occurrence, however the filter value is a continuous one as opposed to the model's discrete hidden state. The components of the filter are not too dissimilar as a result: transition and observation models are employed as a means of state estimation, along with process and observation noise covariance matrices. The assumption is that the actual observed state at a time step evolves from the state occurring at the previous time step, a value determined by the combining of transition probability, the expected process noise experienced by the system, the observation from the system and in turn, its expected noise. The model will predict the state of the system (*a priori*) based on the previous updated state estimate, before then updating based on information obtained through observation (*a posteriori*). Theoretically, the updated estimate is improved with the inclusion of the observation information, which will then inform of the next *a priori* estimate – continuing in a cyclical manner. With the example of GPS positioning, the filter is implemented to combat the fact that the location obtained by satellite is only an estimate (due to noise experienced through transmission, gravitational pull etc.), with the physical laws of motion and knowledge of the noise values informing more precisely of the position due to the compensatory influence they have on the estimation value obtained from the uncertain GPS reading. In the aforementioned case of motion affecting contact sensor readings in behaviour monitoring: the acceleration values obtained through an independent sensor is able to provide this compensatory information so that their readings are perhaps “discarded” or “smoothed” to incorporate the uncertainty of observation.

With the availability and variety of sensors increasing, the number of applications which can benefit is also increasing and as a result the data fusion problem is a growing topic of research for military defence, robotic control, machinery monitoring and medical diagnosis [52 , 53 , 54]. In the survey of [52], the authors identify that fusion is a multi-layered process whereby the input data requires pre-processing before refining and correlation, even before the involvement of any human inputs or decisions. The study performed by Lee *et al.* [43] focuses more specifically on the problems associated with data fusion when applied to healthcare monitoring, in particular the pervasive aspect which the research conducted herein (and as part of the larger project that the research will be ultimately applied to) intends to address. The circumstances and application of their work are very similar (Chapter 3 provides the evidence for this) in that the monitoring system considered is to have both contact and environment sensors taking readings from the user and their surroundings for combining in context to determine the individual’s state. An issue arising from the use of sensor-based observations in probability models directly is their reliability in real-world scenarios. The authors similarly note that certain contact sensors may display motion artefacts in their readings due to intermittent connections caused by friction during motion, therefore requiring that the sensor reading must be appropriately cleaned on the base station post-acquisition utilising prior knowledge of the noise characteristics of such sensors. The previous standard Markov Model technique is dismissed in favour of “modifying a triadic context of hierarchical class analysis” (an approach applied to folksonomy mining [55]) which utilises the available information from the system’s sensors to estimate a user’s state in a spatial and temporal context. There are environmental and body sensors, which through a triadic relation form one set with time and location forming the other two. The notion is that based on the relation of the three sets as exhibited through testing, they can be integrated into a triadic class hierarchy from which knowledge is extracted and a medical professional or carer has the ability to then make a diagnosis or decision based on the information provided. A shortfall of such an implementation in healthcare monitoring is that whilst it may take in temporal aspects of the readings (i.e. time spent in a state) it does not consider the sequential properties of behaviour modelling in the way that the HMM can.

The same fusion notion can be applied to the Programming by Demonstration (PbD) task, which utilises multiple combined sensors from which readings are taken during the learning phase of the device's operation. Typically the programmer performs an action using the device, which is recorded and learnt by the device for an autonomous execution phase. In [56] the PbD device of a "Cyberglove" is used which contains both tactile sensors and the inbuilt bend-sensing strips to inform the system of the grasp shapes and whether or not such a shape is being executed as the result of grasping a real-world object (as opposed to an empty hand). The combination of the two sensing systems distinguishes between two very different states of holding and not-holding, that otherwise may not be determinable using just one of the sensing systems. The Cyberglove is used to record different shapes and sensor values observable from holding and touching numerous objects and an HMM is trained utilising the Baum-Welch algorithm [57] in order to ensure maximum recognition of observations-to-states. This application takes the inputs of the two sensor types and determines intelligently the connections to the state output using the Expectation Minimisation process of the algorithm, therefore bypassing the need to identify correlations and reasoning for the combination of sensor readings producing the states.

A similar such application is developed in [58], with gesture recognition forming a key aspect of the activity recognition of an elderly user on a daily basis. Five basic gesturing states of *come here, go fetching* (sic), *go away*, *sit down* and *stand up* are identified as being observable intents of motion when viewed with a wearable finger sensor. The sensor is an inertial sensor capable of providing 3D acceleration, angular velocity, magnetic and temperature data and connects to a PDA (Personal Digital Assistant) which sends the data to a desktop through a Wi-Fi connection where all state inference occurs. A hierarchical HMM once more is employed to classify the gestures, after the combination - with a 3 layer Neural Network (NN) of observable readings from the finger sensor and two other inertial sensors located on the body at the waist and foot - provides the inputs to the model.

The system can theoretically detect multiple daily activities for the elderly user with the combined body-worn network in a coarse-grain way: *standing*, *sitting* and *lying* are identified as “zero displacement activities”, “transitional activities” are made up of such states as *lying-to-sitting* and *level walking-to-stair walking,* and “strong displacement activities” contains *walking upstairs* and *running.* The NNs classify the waist and foot activity as monitored with the inertial sensor into 3 types (*stationary*, *transitional* and *cyclic*) before the outputs are sent to the fusion centre to integrate the activities and categorise them based on a set of rules indicating the states as described above.

The zero displacement and transitional activities are then more finely classified through the analysis of the inertial sensors and their current observations, enabling such distinctions as *sitting* or *standing-to-sitting* based on the orientation of the waist sensor, for example. The HMM is used to further classify the strong displacement states by recognising the patterns in the time series data from each of the sensors. The fusion of the sensor readings and the coarse- and fine-grained classification processes result in test accuracies of 86% and above, with the HMM being employed only when deemed necessary according to the detection of a possible strong displacement activity. The system as a whole is shown to perform suitably well for the recognition task, utilising the extra two sensors worn at the waist and foot to significantly assist in the distinctions between many subtly detectable states.

With a single, key sensor however, it is also possible to identify activities of a user without the need for fusion or corroboration with other sensors. Curone *et al.* [59] describe an algorithm used to detect posture utilising a triaxial accelerometer. With detectable states such as *falling*, *lying down* and *standing*, the algorithm is integrated into a wearable device where it classifies the user’s activities in an experimental scenario with an above 90% accuracy. Once the required behaviour probability detection systems are developed with Verity using a multi-sensor approach, it would be feasible based on the findings in [59] to assess the reliability of each sensor in isolated conditions perhaps as a means to further provide evidence for a state decision.

The recent survey conducted by Pantelopoulos and Bourbakis [60] specifies typical sensors used in these wearable systems and their intended uses, with attention paid to the fact that "wearable health-monitoring system design needs to take into account several wearability criteria" which includes size, specific application and the security of the data gathered for use in state determining. It is also recognised that there is no single ideal design for behaviour monitoring systems, with each application requiring careful consideration in all aspects of implementation. Rather understandably, a wearable monitoring system must not "hinder any of the user's movements or actions" and radiation concerns need to be accounted for in the design of any system utilising radio transmission. As discussed previously, the pervasive nature of the monitoring itself is noted to be a significant consideration, given that the information gathered by such devices is inherently personal and requires considerable security measures to ensure it remains accessible only to the desired people. Some systems reviewed are identified to be too cumbersome, consisting of too many, large, user-controlled modules to be called "autonomous" [61], whereas others performed suitably well in their evaluation given that they are unobtrusive and require little attention from the user on a daily basis – a property identified by much similar research to be of utmost importance in the design of new passive behaviour monitoring systems.

When the system being used is self-contained and has no means of direct location detection/recognition, the sensing platform must be equipped such that a location can be inferred through other means. The eWatch system proposed by Maurer *et al.* [62] is another wearable device of an unobtrusive form-factor, which tracks the user's whereabouts using a variety of sensors whose readings are compared using the nearest-neighbour method to determine the most likely environment. An innovative development, eWatch uses a microphone and light sensor to record and analyse the lighting and audio conditions in different environments and locations, taking note of the brightness, frequencies of the light present and the intensity of sounds in order to distinguish between most-visited places.

The resultant database comprised of 600 data recordings at 18 locations, with combinatorial results for the light and sound values resulting in success rates of a minimum of 66% and maximum of 98% when the locations were revisited and analysis of the light and audio recordings was performed by comparison with those used in the training phase. The sensors in eWatch are indeed few, but the error rate is proven to be low with the combination of the multiple sensors providing the most accurate readings. This fact goes further towards showing that the combination of many semi-certain results allows for greater certainty in the determining of state or environment.

Home-installed variations have been at the forefront of monitoring-device development for many years, with distributed sensor networks providing the basis for many developed products. The authors of [63] and [64] successfully develop methods of processing the large volume of contextual data from a “Smart Home” environment in which sensors and devices are located throughout – thereby intelligently inferring a belief of a user’s current health state. The system developed in [65] goes further into identifying the health status of the user by measuring observable life-signs while typical daily tasks are undertaken (bathing, sleeping etc.), thereby reducing the awareness of the user that they are being monitored. Those developed in [45] and [23] go beyond ambient monitoring and introduce a body-worn element to further enhance the monitoring reliability; whereas in [66] the authors address the perceived invasive nature of these wearable devices by developing a system which does not necessarily require the user to be wearing a sensor board, yet is able to detect the user’s location based on observations of interaction with the home-installed sensor network. The authors of [67] both highlight and deal with the difficulties of a pervasive system’s reliability, inaccuracy and contextual vagueness through use of evidential fusion based on belief and uncertainty of sensor data.

Whilst acknowledging the documented drawbacks of devices which utilise a wearable or contact-requiring technology, they usually do so in order to ascertain high-accuracy results which can inform the assessor or health care professional of their current health/behavioural state to a more reasonable degree than other monitoring devices. For example, the application in [68] is utilised to assess the severity of a user’s motor complications which arise from Parkinson’s disease – the device remains in contact so as to obtain constant data to inform of their status in a way that pure visual observation would not. Literature surveys [69 , 70 , 71] of recent emerging technology being used for research into health monitoring discuss wearable devices (“healthwear”) that are able to gather sensor data to provide both spatial and contextual information about the user’s environment and their internal health-state in the same way that the system developed here in this research is intended to. The MIThril system of [72] was developed to be a “practical, modular system of hardware and software for research in wearable sensing and context-aware interaction”, consisting of multiple sensing devices which can be integrated within a user’s clothing or placed inconspicuously about their person. The system uses state of the art technology to “hoard” data about the user and their environment before estimating their current state. The central control node is capable of interfacing with mobile phones, cameras, Wi-Fi and even head-mounted displays and the sensor hub has the ability to interface with more hubs in order to expand the sensing capabilities – but a single hub is capable of working with pulse oximeters, respiratory, EEG (Electroencephalography), blood sugar and CO_2 sensors to provide the MIThril framework and software with information suitable enough to determine whether the user is in such states as *standing*, *walking* or *running* with a classifier such as the HMM. The device goes much further than being solely a passive, wearable system though: attaching to the user with medical electrodes to read vital signs as would be similarly carried out in a medical environment – thus removing that pervasive aspect afforded by subtler monitoring devices.

Whilst developed primarily for use in research, a flaw in such a data-gathering method in technology developed for consumers is its invasiveness. The electrocardiogram (EKG) approach of MIThril in determining heart rate is widely used in wearable devices [73 , 74 , 75] yet the need to optimally position and apply 2+ electrodes significantly decreases its usability by a lay-person. Installing heart rate monitors more discretely however is certainly a possibility, with novel ideas proving successful in numerous cases – the review of [76] describes the placement of electrodes and sensors about the home to detect the user during their periods of interaction with those devices.

Such locations include bathtubs, taps, beds and even toilets and therefore eliminate the notion and awareness of the monitoring process. In one such application, 15 thermistors located under the bed sheet and one in the bedroom allowed for significant temperature information to be obtained to inform of the user's body movements during sleep through analysis of the change of temperature distribution. Another application attempted to remove the intrusiveness of the typical lead-approach to EKG measurements, with conductive textile electrodes placed in the pillow and bed sheet as a means of always maintaining contact with the body without the need for direct placement with electrodes, allowing for EKG and respiration readings 97% of the time during the sleep process. The inner wall of the bathtub was identified as an ideal placement opportunity for electrodes to read signs whilst the user bathes, as well as the bottom surface providing the perfect location for a photoplethysmographic (PPG) sensor due to the permanent contact with the users' buttocks. From the signal obtained with the PPG sensor, respiration information is easily obtained using a low pass filter.

Recently, Apple Inc. filed a patent [77] to embed an electrical cardiac monitor into their cellular devices which works to determine heart rate through the user's contact with the metallic housing and another specific location on the device, thus providing the user with the ability to determine when they can and cannot be monitored. Intentions by an organisation such as this for inclusion in devices so widely available provides evidence that product designers are willing to provide the consumer market with the opportunity to monitor their own wellbeing, and that in the ever-changing field of technology, "healthwear" is a lucrative industry.

The communication between the wearable sensors of these devices and even multiple instances of the same devices (nodes) around the environment in a Smart Home is also a growing field of research, with applications utilising numerous wireless standards in order to enable the transfer of information gathered at the sensor nodes to the main processing centre for identification and classification of a state [23 , 78 , 79 , 80 , 81]. RFID, Bluetooth, Zigbee and 802.11g are all typical modalities of communication in the intelligent healthcare monitoring systems, with none yet identified in the literature as more suitable than another for a specific application. As such, Verity uses a wireless protocol optimised for ultra-low power operation of these typical wireless body area networks in order to achieve the same result (Chapter 3).

2.4 The Curse of Dimensionality for Classification

As problems become defined by a richer and greater variety of sources of data, dimensionality can increase. The high dimensionality of data can lead to complex non-linearly separable clusters developing in the higher dimension space which are not easily identified by simple classification methods. The data may in fact contain values in one manifold which allow it to be perfectly separated, however the influence of other dimensions may obscure the required classifiable relationships present in the data.

An issue arises in high dimension data when the distance which separates clusters is too small to enable differentiation, or the data within the clusters themselves is too sparse. The ability to detect outliers in sparse high dimension data is a commonly tackled problem; [82] notes that with increasing dimensionality, it becomes increasingly difficult and inaccurate to estimate the multidimensional distributions of the data points, thus the concept of locality is difficult to define and classification purely according to data density becomes complex and impractical when confronted with the original high dimension representation.

This “curse of dimensionality” is commonly referenced; the greater the number of data attributes (dimensions) the lesser the ability to make sense of the data. This is relevant when addressing the *k*-Nearest Neighbour (*k*NN) function [83] due to the fact that with a higher dimension, standard Euclidean distance functions lose their usefulness and so clustering with such methods becomes less accurate. There are essentially 4 main problems which relate to the “curse of dimensionality” and the increasing number of attributes [84 , 85 , 86]: Optimisation becomes difficult, relative distance between extreme points converges to 0 (discrimination between nearest and farthest neighbour becomes poor), dimensions become “noise” - given that their relevance to the data may be little - and some dimensions may even “exhibit correlations among each other”, thus becoming redundant.

A dimension reduction technique capable of separating nonlinear clusters from the high dimension results in a lower dimension dataset on which successive linear-based operations can be performed – for example a simple binary classifier trained on the newly linear separated data is able to then identify an untrained data point’s membership to a cluster.

Within the dimension reduction field are a multitude of techniques for dealing with nonlinear, high dimension data, with many sharing basic underlying principles to reach the lower dimension representation of a complex nonlinear data set: Sammon’s mapping, Isomap, Curvilinear Component (and Distance) Analysis all seek to replicate similar distances between points located in a high dimension after placement in the lower dimension, by a means of gradient descent or iterative error reduction methods.

Sammon’s mapping [87] is considered to be one of the first described nonlinear dimension reduction techniques, simply aiming to minimize the error function representing how well the configuration of points in the high dimension (*n*-space) fits the projection in the lower dimension (*p*-space):

$$E = \frac{1}{\sum\limits_{i<j} \left[d_{ij}^{*} \right]} \sum_{i<j}^{N} \frac{\left[d_{ij}^{*} - d_{ij} \right]^{2}}{d_{ij}^{*}} \tag{1}$$

Where i and j are two vectors found in *n*-space, with a distance d_{ij}^{*} between them (typically measured as Euclidean, or “straight-line”). The error function is reduced by a method of steepest descent, moving the points in the *p*-space incrementally based on the calculated error such that the result when re-calculated (d_{ij}) is the minimum value at the final point placement in the lower dimension.

Isomap [88] works in a very similar manner by identifying distances between points according to a geodesic measurement over Sammon’s basic Euclidean, pointing out that the shortest path distance between far apart points on the manifold can appear deceptively close in the high dimensional space when measured with a straight line distance. The first step in implementing such a technique is in the identification of which points on the manifold are neighbours based on the distances between the pair of vectors

i and j, before their connection to each other through a metric of either *k*NN or interconnection based on their falling within a pre-set radius. These pairs are then stored in a graph with edge weights equal to their distance d_{ij}. The geodesic distance calculation is an estimation of shortest distance between all point pairs on the manifold, which is computed using a simple graph search algorithm returning weights based on the between-pair distances and is stored in a matrix of geodesic graph distances $D_G = \{d_G(i,j)\}$.

As with Sammon's mapping, the projection error in *p*-space is minimised such that the manifold's geodesic distance is preserved as Euclidean distance according to:

$$E = \left\| \tau(D_G) - \tau(D_p) \right\|_{L^2} \tag{2}$$

Where D_p is the matrix of distances between vectors i and j in the lower dimension, and $\|A\|_{L^2}$ the matrix norm $\sqrt{\sum_{i,j} A_{ij}^2}$. τ "centres" the matrices allowing for the eigenvectors and eigenvalues to be computed and the projection coordinates chosen according to top *p* eigenvectors of the matrix $\tau(D_G)$.

Despite improving on the functionality of the Sammon Mapping of the high dimension to the lower dimension to better represent points at different ends of the manifold, what is recognised by Lee *et al.* [89] in their creation of the Curvilinear Distance Analysis (CDA) [90] is that Isomap's random selection of point pairs results in the lower dimension manifold appearing as a torn and stretched representation of the original data set, with the initial distribution being replicated incorrectly. Essentially, the Isomap procedure works to preserve long distances in the manifold by comparison with shorter ones. Employing vector quantisation to obtain the initial prototypes for pairing yields better results, without the problem of tearing; however representation of 3D manifolds still remains unsatisfactory when neighbours are identified at opposite ends of the manifold – the resultant projection typically appears "crushed" and stretched due to the "parasitic link" in the centre of the manifold which is unable to be overcome with Isomap. The Curvilinear Component Analysis (CCA) [91] on which Lee *et al.* base their scheme on, also attempts to find a lower dimension representation of a manifold which "minimises a cost function based on interpoint distances in both input and output spaces".

A new cost function is created in CCA which is capable of unfolding strongly nonlinear structures, and in its minimisation a significant speedup in projection is found. This function is not too dissimilar to Isomap, which takes as inputs the Euclidean distance between vector quantisation-identified point pairs, $d_{i,j}^{n} = d\left(n_i, n_j\right)$:

$$E = \frac{1}{2}\sum_{i}\sum_{j \neq i}\left(d_{i,j}^{n} - d_{i,j}^{p}\right)^2 F\left(d_{i,j}^{p}, \lambda(t)\right) \tag{3}$$

At its core, the function aims to reduce the error between the high dimension distances ($d_{i,j}^{n}$) and the lower dimension representation ($d_{i,j}^{p}$), with the addition of the extra bounded and monotonically decreasing function $F\left(d_{i,j}^{p}, \lambda(t)\right)$ which enables the local topology to be recreated more reliably – the step function can be used to identify points that fall within a chosen neighbourhood, resulting in local representation being favoured in the error calculation over larger distances. This neighbourhood is ideally decreasing so that the projection remains suitably representative of the original manifold.

The CDA scheme further enhances CCA by eliminating the Euclidean distances between pairs ($d_{i,j}^{n}$) from the calculation, which is identified to be a problem in manifolds such as a spiral - which would link neighbouring points through the manifold rather than around it and result in further projection errors. The replacement is with the curvilinear distance, $\delta_{i,j}^{n}$, which is a sum of "Euclidian lengths of all links in the shortest path from centroid i to centroid j, provided there are no 'shortcut' links", resulting in a distance which doesn't cut through a manifold.

$$E_{CDA} = \sum_{i=1}^{N}\sum_{j=0}^{N}\left(\delta_{i,j}^{n} - d_{i,j}^{p}\right)^2 F\left(d_{i,j}^{p}\right) \tag{4}$$

The projection is a true "unfolding" of the high dimension data set, with the curvilinear distances "flattened" to Euclidean in the lower dimension, which has significant benefits for nonlinear projection requiring visualisation of the manifold.

The extensive (yet non-exhaustive) survey of [92] highlights key techniques utilised in the dimension reduction arena, distinguishing the types of dimension reduction methods applicable to both linear and nonlinear data. The theory outlined is that in high-dimension data sets with both linear and nonlinear properties, not all variables measured are “important” and therefore become obsolete from subsequent operations, allowing for their removal by dimension reduction. The authors consider Principle Component Analysis (PCA) [93] as the best linear dimension reduction technique, as it seeks to find a lesser number of linear combinations of the original high-dimension variables with the largest variance, resulting in a reduced dataset described by the first few combinations. Random Projection methods are also considered, whereby the high-dimension data is simply projected into a lower dimension space in a randomised nature and clustered before the next processing step is administered. Results are shown to be more suitable than PCA for speed, yet a slight loss in accuracy is expected.

A combinatorial method for dimension reduction is described in [94] which applies itself to speech recognition using HMMs. Linear Discriminant Analysis is employed to find a low dimensional projection of data such that the variation between and within classes is largest, before the mean parameters used in an HMM for pattern recognition are found for the data with Reduced Rank Estimation.

Neural network based classifiers can also be applied to nonlinear high dimension data in order to identify the nonlinear relationship and classify successive points according to the properties of the data set. The problem with such approaches is their need for trial-and-error in the choice of the number of layers, neurons and iterations if the data set is not considered “standard”. The end result may also be impractical if the determined “solution” for a parameter has actually fallen in a local minimum, thereby producing false results and reducing the sensitivity of the classifier.

With multiple sensors used in the behaviour monitoring task, the dimensionality of the resultant observation set is considerably high, especially when the system can be expanded upon to incorporate more sensors throughout use – thus instigating a significant need to identify correlations between sensor readings which make up each dimension as a means of classification. With such physiologically observable signals as heart rate and acceleration or temperature and acceleration making up one manifold of a data set, their often nonlinear relationship requires overcoming through dimension reduction in order to better enable in the determining of state probabilities which are subsequently used in sequence determining models like the HMM.

2.5 Instance Based Learning

Instance Based Learning (IBL) takes directly sampled data from any system and constructs a hypothesis regarding similarity without the need to generalise a model based on the often high dimensional and nonlinear data – resulting in a classifier which makes its decisions according to the actual data and hypothesis rather than an abstraction formed from its analysis. Through training, data instances are stored in some form of memory which is then accessible for subsequent classification operations, where a query is submitted and compared with all trained values according to some distance metric in order to ascertain its membership to the encoded classes.

With the often-extensive amount of data available to (and supplied by) passive behaviour monitoring systems due to high dimensionality and nonlinearity, the time taken to postulate a relationship between state and observation can be significant and require multiple rounds of training before an adequate hypothesis is formed. The data orientated approach of IBL would remove the need for model generation and therefore show promise for speedup during the classification process.

It is commonly accepted that the genus of and starting point for IBL algorithms is the simple *k*-Nearest Neighbour classifier [95]: saving training instances to some data structure such that other instances may be compared distance-wise with those local data already classified to return a possible containing state for the new instance [96].

The Nearest Neighbour algorithm implementation can be considered in several ways. Traditionally, the literal interpretation is to assess the *k*-Nearest Neighbours for their class membership, with the majority class being considered the winner to which the query point is attributed [83]. Another interpretation can be extrapolated to operate on a similar basis to fixed radius nearest neighbour searching [97], classifying points according to a majority vote within a radius: rather than assessing a pre-defined number of points as in *k*NN, a pre-defined radius, ω, is determined within which the class of all points are tallied and the majority is then considered to be the winner (in a tie-break situation, the radius might be expanded).

$$c(q) = \max_{0 \le j < J} \left[\left| p_x = c_j \right| \right], \text{ where } \left(q - \omega \le p \le q + \omega \right) \qquad 0 \le x < n \qquad (5)$$

Equation (5) illustrates the selection process during which the query point q is assigned to the class c. The winning class is that which contains the majority of trained data points p within the given radius ω.

With a *k*NN algorithm, the data may have a very high dimensionality resulting in the requirement for an expansive storage space in which to hold all data accurately. Similarly, the increase in the amount of data to parse in order to determine those points within close proximity to the queried instance has a significant effect on the speed at which the algorithm can operate. With IBL methods that operate on dynamically updating data streams the problems can arise quickly and more noticeably than on more static data [98] – but the mining operations occurring on the data sets also have a significant bearing on the tolerance of the speed at which they occur. Pattern [96 , 99] and anomaly detection may vary in their importance to the application [100 , 101 , 102 , 103 , 104]: from a machine-health point of view the sooner a problem is detected after examining multiple signals then the sooner the fault can be corrected (similarly with network intrusion detection [101]), however in the case of financial data or worker-behaviour analysis an anomaly which would indicate that future behaviour required modification may only become apparent over a longer period of time and so instant calculation and processing is less vital [105 , 106 , 107]. For operations which attempt to classify data in a continuous stream there is typically a need to function in real- or close to real-time such that the data itself is then able to be used in subsequent look-up operations for determining another instance's classification.

There are a number of instance-based classification algorithms that address the speed optimisation issue for real-time operation in a number of fields. The IBL approach to classification in data streams is considered in

[98], highlighting the key requirements that learning algorithms must be highly adaptive and deal with the time-varying concepts that come with a continuous data stream – with the IBL method considering only a small subset of the stream rather than the entire universe of samples which would result in heavy computation.

The memory created in this developed solution for data stream classification is autonomously updated, taking the arrival of a new sample as a reason to re-assess the relevance of all cases currently stored based on a neighbourhood metric. A set of samples within a neighbourhood of the new arrival are considered as candidates for removal, with the exception of those most recently added due to the fact that noisy data may be difficult to distinguish from the beginning of a concept change. In the case that the class of the newest sample is the most frequent among those in the candidate set, then all samples of a different class within that neighbourhood are discarded, maintaining the memory's relevance to the data stream.

RIONA (Rule Induction with Optimal Neighbourhood Algorithm) combines rule induction and IBL to inform of a state decision based not on an entire rule base, but on a restricted set within the neighbourhood of the test case [108]. Utilising *k*NN the decision is made in a vastly accelerated manner, with the solution reportedly proving more suitable for *k*NN problems and rule based classifiers. The algorithm considers only the samples covered by the rules matching the test case falling within a specified neighbourhood. A rule is constructed based on the test case and a neighbouring sample, then checked for consistency with the other samples in the neighbourhood. If the new rule is indeed feasible given the samples, the original sample is added to the support set for that rule.

Recognising that the brute-force approach to a nearest neighbour search is often the best when dimensionality is high, Toyama *et al.* [109] propose a method to greatly reduce the search time and return a correct set of *k*-Nearest Neighbours with a high probability, exploiting the marginal distribution of the nearest neighbours in low dimensions and the fact that "a very close pair in the original *m* dimensions is also close in a few *l* dimensions in high probability". Utilising a PCA approach, only the first few values of a projected data point in a lower dimension are checked, which, with a predictable ratio, reduces the search time experienced if executed in the original dimension.

As highlighted by Toyama *et al.*, for large data sets with high dimensionality (d), searching through n instances of a data set in order to determine those within the closest proximity can take an extensive amount of time, given that all pairs require evaluation using a distance measure such as Euclidean or Hamming. When considering the traditional nearest neighbours algorithm, the time complexity is deducible to be $O\left(dn^2\right)$, when sorting is employed to determine the *k*-Nearest Neighbours.

Locality Sensitive Hashing (LSH) [110] provides adequate means to speed up the process of nearest neighbour searching, overcoming the above issue by storing the data in another variable-tolerance, compressed format which is easily searchable and requires only simple look-up operations to determine possible immediate neighbours which can then be subjected to a linear search to directly find the *k*-Nearest Neighbours. The principle behind LSH is to hash the sample data in such a way that the probability of two points hashing to the same location of a 1-dimensional data structure is higher for objects that are close to each other than for those that are further apart. Given a subsequent query point, the most similar points in the database are desired to be found so that the class of the unknown value can be inferred as a result. With the high dimension data processed with an LSH function, the database is easily searchable without the need to employ a distance metric in the initial look-up of all samples and queries, given that processing occurs in the storage stage – reducing the overall computation from a distance search of all points, to a select few.

The process of simple LSH is thus: identify the dimensionality in which the database exists, and randomly select a family of vectors from a Gaussian distribution each with a dimensionality equal to that of the original data space. The dot product of the point in the database and each of these random vectors is then found, before the result is divided by a value termed the "bucket width" – which then when subjected to the floor operation results in an integer value known as the "hash location" in the 1-dimensional structure. At each respective hash location/bucket the original database vectors are then stored, in theory with each vector in one location sharing similar dimensional values. The bucket width essentially acts as an upper radius for the nearest neighbour operation, with nearby vectors in the high dimension space falling into the same hash location if the bucket width is appropriately sized. Increasing the width will increase the number of points that fall inside it, thus reducing the size of the final database. The trade-off in the implementation of an LSH is between having a larger table with a smaller final linear search to locate the nearest neighbours, or in having a more compressed table with more vectors to consider [111].

Once a query point's location is identified with the same LSH functions as used to encode the original database, the vectors contained within the hash locations form the searchable list for the subsequent standard nearest neighbour operation; a speedup in identification of the *k*-Nearest Neighbours is therefore observed due to the decreased number of pairwise distance operations carried out on all data. The fact remains however, that whilst the database has been optimally indexed to allow for faster query speed, the sample storage requirements remain high as no compression occurs beyond the dimension reduction on each point possibly chosen to be employed at the active hash locations [112].

Whilst IBL algorithms have multiple advantages over parametric and model-based algorithms, especially in the storage of new, unseen instances (other algorithms would typically require complete re-examination of the data set in order to be wholly inclusive of the new data points where IBL methods simply "insert" the new data instance without disrupting any previously determined hypothesis), with large and sparse data sets there comes a problem in the storage of all instances. This is especially the case if future instances not currently contained within the data structure require insertion - thus increasing the size of the structure to a degree which may result in the decreased efficiency of the lookup mechanism or even cause overflow of the storage space. Taking influence from the LSH data structure, involving hashing and indexing of high dimension vectors: a solution appears in the form of a Bloom Filter, which is capable of holding an extensive amount of data in a limited space in order to facilitate a simple checking of the data point for membership to a state.

Originally developed by Burton H. Bloom [113 , 114], a Bloom Filter is a simple data structure which holds information regarding an element's membership to the data set that the filter represents, storing it in a compressed format as governed by the multiple properties of the filter. The filter itself is actually a finite-length bit vector (in the original implementation; subsequent filter designs allow for dynamic resizing [115 , 116]) which in its unpopulated state contains only zeros, or null values. The zeros are set to ones to correspond with the outcome of a series hash functions when an input element/vector is presented to them. The input element will then have a series of corresponding values set to one in the array which when queried in a later operation indicates that the element belongs to that filter and therefore to that data set. With more hash functions, more activated values can appear in the filter, which although ultimately increases the certainty of the element's membership when the corresponding values are checked for activation, in standard Bloom Filters increases the false-positive rate due to a higher collision rate. In practise, for each application there are an optimal number of hash functions as discussed below. The flaw of a Bloom Filter in its ability to produce these false-positives – incorrect "hits" which indicate an element is a member of that set when in fact it is not – is considered an acceptable trade-off against its efficiency. This "flaw" actively influences how to implement a Bloom Filter for a specific application, with multiple factors requiring consideration in order to determine the most acceptable false-positive rate before any learning or memory operation can occur if accuracy in classification is required over compression rate.

A Bloom Filter is capable of reducing a data set of n instances of any size/dimension to a simple array consisting of m bits. The lookup to determine another instance's membership to the class's representative array requires only a check of the h hash function's bits in that array, where graph search algorithms may require processing n elements for a returned membership value. These values of n, m and h are reliant on each other and all optimisable depending on the aim of the application (i.e. compression percentage, recollection accuracy etc.). Their combination influences the rate of false-positives for the given data set in the filter:

$$\text{Bloom Filter}\left(P_{fp}\right) = \left(1 - \left[1 - \frac{1}{m}\right]^{hn}\right)^{h} \approx (1 - e^{-hn/m})^{h} \tag{6}$$

Provided that the number of hash functions is set at its optimum (7) to minimise the false-positive rate, the minimum false-positive rate for a filter with the above parameters resolves to be as in (8), relying solely on the number of hash functions used for encoding:

$$h = \frac{m}{n}\ln 2 \tag{7}$$

$$P_{\mathrm{fp}} = 2^{-h} \tag{8}$$

The Bloom Filter implementation described by [117] deals with the matter of privacy as identified previously in applications such as passive behaviour monitoring, suggesting collaboration with a *k*NN classification method as a means of allowing the sharing of data whilst preserving its privacy. Real data values are hidden by encoding the sensitive information with hash functions and storing them in a Bloom Filter which is accessible by the querying party. The results received from a query won't compromise the original data privacy, as the hash function operates one-way and is typically rather difficult to decode when a good hash is employed. A hash is received as part of a query and the *k*NN to that obfuscated query are selected from the database with results returned to the requester without their need to see the actual information database.

In a similar way, Bloom Filters passed as messages in computing protocols are a means of reducing the transmission size of large databases. In the application of web cache sharing, a proxy will hold information regarding which sites it holds in its database which is in the form of a Bloom Filter. To reduce the number of transmissions over a network, proxies will periodically broadcast their Bloom Filter rather than the entire site list, resulting in each proxy maintaining multiple Bloom Filters which they can query to ascertain the presence of a site on another proxy [118]. Mitzenmacher [119] introduces the concept of a Compressed Bloom Filter which can subsequently improve performance in the transmission and web traffic reduction application, reducing the number of bits broadcast and the computations per query at the expense of an increased processing overhead in the end machines required to compress and decompress the data.

Identifying that when Bloom Filters are the objects transferred between proxies, the number of hash functions to be used must be optimised for the size of the transmission and not the size of the initial filter or its contents: the value is chosen such that the probability of an entry in the filter is 1 with a probability 0.3, rather than 0.5 as with an uncompressed filter which is optimised for storage size. This reduction in probability allows for a smaller value of m which is then easily compressed to improve transmission size. It is found that the number of hash functions minimising the false positives without compression maximises the false positives with compression, therefore indicating that for compression and transmission of the Filters in the manner outlined in [119], compression will always decrease the likelihood of false positives. This does however introduce the need for increased processing in the compression and decompression of the transmitted filter, and therefore proves unfeasible for behaviour monitoring applications required to operate at high speeds for real-time classification. Such an implementation would be recommended though when transmission is required between the monitoring device and a storage medium perhaps after a period of monitoring has elapsed, reducing the packet sizes and power consumption as a result.

Combining the LSH theory with Bloom Filters, Mitzenmacher and Kirsch further develop a Distance-Sensitive Bloom Filter [120] in which the random hash functions of a standard Bloom Filter are replaced by LSH functions, allowing for queries to identify locations containing possible neighbours of the same class based solely on their distance from the initial lookup location within the filter. The modification is shown to not enhance the performance of a standard Bloom Filter in basic lookups, yet by limiting the goals of the scheme it shows potential for numerous applications in networking and database fields – a fact employable in behaviour monitoring where an observation can be queried to a database to ascertain its membership and similarity to points of the same class, with the added bonus that the storage structure is in a compressed format and thus allows for expansion to include further untrained points if necessary.

2.6 Fault and Anomaly Detection

With the detection and recognition of behaviours there must also be an anomalous behaviour detection scheme in order for the device to become truly useful in operation. Many care systems relay state information and behaviour sequences to a central hub located in the care home, or to an online webpage that can be monitored remotely by the caregiver from anywhere and at any time. The drawback of such systems is their requirement for human involvement at what can be seen to be the most critical time in the monitoring: that of an irregular reading or behavioural event. If there were an inclusion of an autonomous fault detection scheme to assess the validity of the data retrieved during operation before the caregiver or monitor were informed, the faults and anomalous behaviour could be isolated at an earlier stage before any danger escalates.

Anomalies can be seen in the dynamic data as it is computed, i.e. a monitored behaviour should not logically occur after the previously seen behaviour; or in data obtained over a long collection period, whereby the current behaviour pattern does not fit with that usually witnessed. In the case of the behaviour pattern mismatch, there is usually a threshold in place which determines whether the difference in behaviour has reached a point where the user could be experiencing problems. When the monitoring is periodic - in the sense that the observations determine a sequence of states in an hour or maybe over a day – the monitoring process can reference previous collected data to ascertain a probability of anomalous activity. For a reliable and logical monitoring service, the data would be collected over a 24 hour period given that the majority of tasks experienced are repeated to some degree on a daily basis. A work by Virone, Noury and Demongeot [121] employs a smart home system to monitor a subject over a simulated 70 day period and obtain average values for room habitation on a daily basis. The average acts as a probability of room occupation, against which the actual room occupation at an instance is compared. With thresholds set for under and over-occupation, should the current value deviate from the expected then a suitable alarm is triggered. Parameters are defined for serious and minor deviations, with checks made with the user to ascertain whether an alarm should be triggered. The scheme uses simple statistical analysis to govern the detection and the results show effective application. It is shown that it becomes easier to detect anomalies in data which is more predictable and given context information an event which has a low probability of occurrence is classed as an anomaly [63]. An issue arises however in the distinction between weekdays and weekends (as noted by the authors) as the behaviour pattern may differ on such days. For any monitoring product of this type (daily behavioural traits, rather than sequential behaviours), this is a significant consideration [9].

An analysis of room occupation and activity data could provide information about the day and nature of the activity itself. These monitoring systems usually take the form of smart homes which incorporate numerous built in sensors to observe various areas and objects in the home. Helal, Cook and Schmalz [122] developed a system using the already-equipped CASAS Smart Apartment at Washington State University which was able to provide detailed information about the user regarding their activity schedules and behavioural patterns, highlighting specific instances where a task was incorrectly executed or forgotten. The system used techniques including Markov modelling to recognise behaviours and the experimental results showed that 98% of activities were correctly identified.

Although applied to a subject who is still active at work, the principles addressed by Barger, Brown and Alwan [64] are applicable and transferable to elderly behaviour monitoring. Their observation is that a worker with variable work days will upon observation have noticeable deviations in “normal” behaviour on a day to day basis. This could affect an alarm system unfavourably given that a detected unusual behaviour may in fact be due to the subject being at work and using various rooms of the house at different times and by varying amounts. The data obtained can however be analysed to logically identify work days by the activity in given rooms, and the time at which each activity occurs. Once such an operation is applied, the issue of unusual behaviour is addressed. The viewing of room activity data may return clusters when averaged for day-to-day over a long observation period, but there may be activities which do not fit with the usual pattern. This may be due to simple tasks requiring unusual trips to rooms, or tasks occurring which may have taken longer than expected. These events are classed as random and are therefore not fully considered by the behaviour detection scheme. The personal monitoring of the user through wearable devices may require such considerations when the behaviour of the user deviates from the normal. The overall behaviour pattern may vary very little over a long period, but on a daily view there may be some activities which occur only for very short periods. In such circumstances, the temporal considerations expressed in [35] come into effect. Whilst the observed behaviours may be possible in a shorter timescale, they must be queried for correctness when the larger sequence is considered. Repeated observations of these behaviour instances may indicate problems with the individual which could range from dementia or Alzheimer’s when consistent repeated room occupation is observed, to Narcolepsy with Cataplexy [123] or Sleep Apnoea when a body sensor system is utilised and consistent repeated body stillness is observed.

The task of anomaly detection in behaviour models can even draw on methods implemented in traditional fault detection of industrial systems. With a behaviour model, there are similar aspects which can be translated to the industrial system. The inputs to an industrial system may be equivalent to the observation sensors used in the behaviour system with the outputs, instead of being a continuous signal typically providing information regarding the health or productivity of the machine in question, being the behavioural states determined from the sensor observations. In industrial fault detection the actual values for inputs and outputs are usually unknown, so sensors are used at the input and output to provide measured values and the fault detection is treated as an observation problem [124]. With the behaviour model the actual inputs are known as they come directly from the sensor readings of the subject, and the outputs are the states determined by the process. The process is the only thing which can vary in healthcare monitoring schemes, with some opting to employ the probabilistic sequence models [33 , 125 , 126] while others use fuzzy techniques and fusion to combine the raw sensor readings [43 , 127]. Due to the similarities in the system topology, faults which occur in the industrial system in actuators, inputs, controllers and the process itself can be applied to the behaviour system. The fault occurring in an input may be due to the failure of a sensor;

the actuator fault may be seen as a problem with the user; and a controller or process fault could arise due to incorrect behaviour model parameters. The scheme must therefore be capable of not only detecting a fault but identifying the type experienced in order to correct the system or trigger the appropriate alert scenario. Despite the suitability of the application in behaviour monitoring, a literature search returns few relevant discussions on the topic.

An extensive survey by Chandola, Banerjee and Kumar of anomaly detection methods [102] discusses the many developments in the field and the applications to which each methodology is best suited. A point made in the survey of significance to this research is the distinction between *anomaly* and *novelty* detection. Where an anomaly can be seen abstractly as data which does not fit with a normal observed pattern, a novelty is first detected as such but then reasoned to be part of the new normal behaviour and is then incorporated into the normal view for future behaviour comparisons. In the case of the monitoring system, this may be prevalent in the early stages of implementation where the learning of the user's behaviour patterns occurs at its highest rate. The default normal behaviour may be a variation for a user, yet the initial anomaly detected may actually be part of their usual pattern, requiring a test by the system to ascertain if this is the case before proceeding with the consequential action. This is an issue faced by outlier detection in computer networks, where the dilemma for effective detection stems from the need to have large amounts of sequential data in order to successfully formulate a belief of normal behaviour [101] before adequate detection techniques are applied. Again, the need for contextual awareness becomes significant when evaluating a sequence for anomalousness in computer networks. A process U viewed amid some other processes may be normal yet infrequent; however when process U is seen to occur after processes S and T the situation signifies a network attack. In many ways this reflects the temporal problems with human behaviour monitoring discussed earlier.

Fine [100] proposed a clustering method for use with behaviour anomaly detection whereby the observations used are collected over the span of a 24 hour period. The data is assessed as a whole and compared with previous data which has been termed normal. The observation is taken over such a period due to the fact that while behaviours may be cyclic on a daily basis, their timings may differ throughout the day. The example given imagines that the subject remains asleep for half an hour extra on one day and therefore causes all subsequent actions to occur later than usual. If the single instance of an action were to be compared with a previous day at the same time, the distance between the two would be great. However, if the actions were viewed as a whole throughout the day, it would be seen that there was a shift in all actions by the same amount based on the observations surrounding it. The method computes the Hamming distance between two observations, cycling through the 24 hour data and increasing or decreasing the weights as similarities occur. The Levenshtein distance is also calculated to assess the similarities, thereby allowing for the time difference which may be apparent. The experimental results are promising, yet after the clustering of the observations for anomaly detection both distance computation methods return similar results. The underlying principle of analysing observation distances as a whole appears a reliable aspect to include in a behaviour monitoring implementation.

It is not such an abstraction to recognise the relevance of the Hidden Markov Model as used for behaviour monitoring for application to error and fault detection, if the hidden states to be discovered are perhaps outliers and anomalies trained on observations which indicate such a fact. Joshi and Phova attempt to classify network traffic as either an “attack” or “normal” behaviour by building a predictive anomaly detection system based on the HMM [128]. Observations are labelled according to values obtained during a TCP session, with a certain combination of observations indicating whether the TCP session is indicative of an attack or not.

Jecheva [129] similarly discusses the application of HMMs for intrusion detection, noting that there are differences between misuse detection and anomaly detection. Misuse detection focusses on alarm generation based on attack signatures that are known to indicate intrusive activity and as a result are ineffective when presented with unknown attacks. Anomaly detection however deals with the creation of an expectation profile of standard usage, where any deviation from this “normal” is considered to be an anomaly and thus triggers an alarm. In the HMM implementation documented, the latter scenario is facilitated through use of a threshold acting on the Viterbi algorithm used for sequence determining. Once the observations regarding system calls are processed by the Viterbi algorithm to determine the user activity sequence, the mean of all probabilities of the most probable state at each time step is calculated and compared to a user-defined threshold. If the mean activity probability appears less than the threshold considered indicative of a normal activity sequence, the intrusion detection sequence marks the process as an anomaly.

This modification and use of the inherent HMM ability for sequence determining is not a widely adopted approach in the literature, as the model itself is typically used with the hidden states directly describing different forms of normal or abnormal behaviour as inferred through observations. Utilising the probabilities calculated at each stage to ascertain a likelihood of abnormality is therefore an expansion of the HMM's typical use which doesn't impede on the functionality of the sequence determining; instead for the purposes of behaviour monitoring it may in fact prove useful where the primary aim is to determine a state, which *consequently* informs of abnormalities with a sequence.

2.7 Summary

The review of the current state of the art in the behaviour monitoring and assistive care fields informed of the multitude of different commercial products and research projects concerning ambient assisted living, with a number of provisos and considerations learnt and drawn regarding possible implementation of the Verity system: from size and form factor, to operational requirements including the ease of use where the elderly are concerned and aspects regarding the intrusiveness of the monitoring device.

The Hidden Markov Model was identified as the most appropriate model for use in passive behaviour monitoring due to its probability characteristics and decision ability based on observations obtained through distributed sensors, with a number of current and past applications providing evidence that such a model can be successfully implemented for the outlined monitoring task.

From identification of the Hidden Markov Model, it was observed that multi-sensor fusion is required in order to more adequately perform the classification with a distributed sensor system; Kalman filtering was identified as an ideal method to combat noisy observations through the employment of multiple sensors, and fuzzy and discriminant fusion techniques were compared as a means to discover the most appropriate for application in Verity.

The issue of dimensionality was addressed when behaviour monitoring applications utilise rich and varied data sources, resulting in the requirement to overcome the need for classification in high dimensional spaces: which often provides a model with imperfect information due to differences in the weighting of each dimension of data. Key dimension reduction methods of relevance to similar data to that obtained through behaviour monitoring were reviewed; identifying that manifold learning and unfolding proves most suitable in applications aiming for visualisation of data, as well as succeeding in the generation of an element of linearity in the lower dimension from a higher dimension nonlinear source to facilitate the classification task.

In the case of large datasets, nearest neighbour Instance Based Learning algorithms often fail in practicality due to large storage and computation requirements. Implementation of an IBL method for the behaviour monitoring application was suggested as a means of speeding up classification of sensor data before use in the HMM, without the need for implicit dimension reduction beforehand. Database storage with a Bloom Filter was described as a means to overcome the size problem associated with large quantities of high dimensionality data, with literature reviewed detailing successful usage in similar IBL applications.

Finally the issue of anomaly detection in behaviour monitoring was covered, with similarities drawn between the manufacturing industry, networking and system anomaly prediction applications in their applied methods for finding faults. Current monitoring systems were described which utilise temporal aspects of state detection to identify errors, and the HMM was discussed as being a more than suitable means of detecting anomalies with applications of some schemes in the network and machine fault diagnosis fields providing evidence of the model’s viability for such a use.

Chapter 3
Verity: Hardware and Preliminary Operations

The device on which this research is primarily intended to run and is being developed for is portable, unobtrusive and designed to be self-contained such that all operations can be carried out with no further programming input from either the user or the system designer once the automated algorithms are implemented. The complete system – base station and direct monitoring device - is called Verity, and will continue to be referred to as such throughout this work. Verity was developed alongside this research by other members of the project and this section details the hardware and the operations which execute in the initial stages of the behaviour monitoring process on the system before the intelligent identification algorithms developed in this research receive their input data.

3.1 System Hardware and Interaction

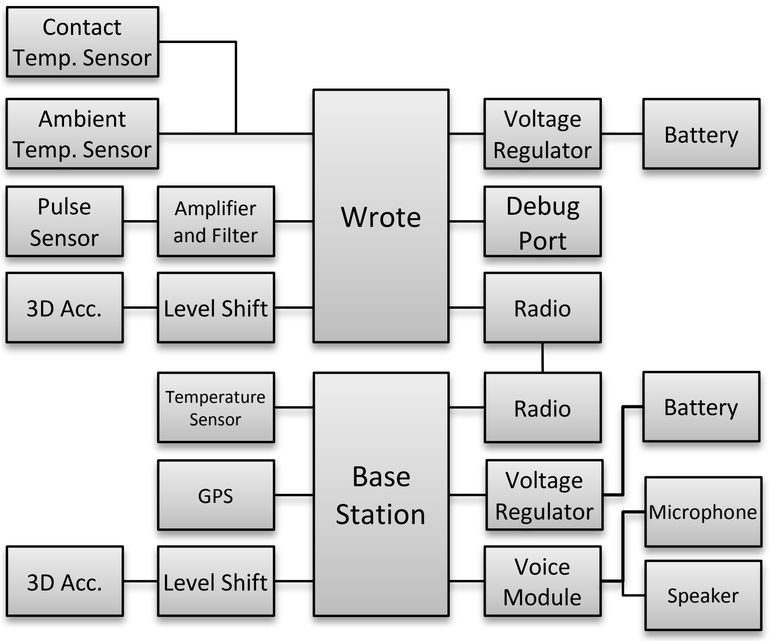


Figure 1 Overview of the structure of Verity and interaction between the on-board modules

In Verity we develop a mobile sensor platform, designed to deliver enhanced and integrated personal monitoring and communications, with a primary target market of elderly users who find themselves with a requirement for a level of care only possible through round-the-clock observation, which would otherwise prove costly or impractical if provided through direct contact with an in-home carer.

Verity exploits state of the art technology to enable an efficient and unobtrusive method of data gathering from a user through a combined sensor approach, with the functionality of a voice-operated mobile phone embedded within the system as a means of communication with both user and care provider when intelligently-triggered alarms or alerts require external confirmation or attention.

Verity is a system comprising two separate components: the base station and the direct monitoring device (Figure 1), each in turn consisting of specialised modules designed for various aspects of the behaviour monitoring task. The base station is constructed to resemble a mobile telephone-type device, being no bigger and weighing no more than a standard mobile telephone. Its primary function is in gathering secondary data and processing the sensor observation information received from the direct monitoring device. The direct monitoring device is intended to be as unobtrusive as a wrist watch, therefore is sized for inconspicuous placement on the wrist and named the *Wrote* (*wr*ist m*ote*). It is designed to be worn in direct contact with the radial artery, such that the on-board sensors are optimally placed to obtain the primary readings from the user.

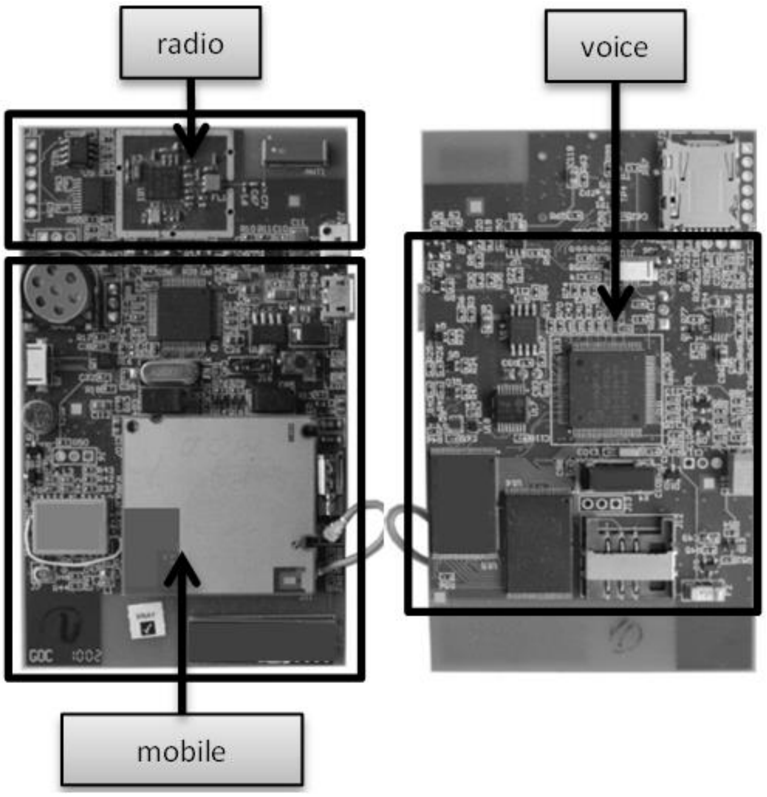


Figure 2 Verity's base station layout; the left and right images show the top and underside, respectively.

As ambient assisted living devices – and more specifically, passive monitoring systems – are by their very nature meant to be inconspicuous, the design of the Verity is therefore an important factor when developing a new product aimed at this market; with Verity modelled as it is, this remit is satisfied.

The developed base station (Figure 2) is primarily concerned with processing data gathered by the Wrote and the subsequent decisions to be made based on this data using the algorithms developed in this research to obtain a state belief. It houses the main intelligence of the system. Figure 1 and Figure 2 show the contents and layout of the base station, with the 3 key areas of inter-device communication (radio), user communication (voice) and alert communication (mobile) highlighted. Within these areas are contained relevant components for actuation of the device's purpose, such as a GSM module (the Quad-Band-capable Siemens MC55i, from Cinterion) with SIM card interface for mobile telephony, a vibration motor for haptic feedback of a call alert (JinLong 4FC1B1301781), a GPS receiver (Fastrax IT321) to provide accurate information regarding the location of the device/user, a microSD memory card slot for storage of voice information for the voice recognition chip as well as a speaker (Star Micronics' SCB-13A) and omnidirectional electret condenser microphone (WM-63PR from Panasonic) to enable multi-way communication between the user-and-system for the mobile phone and voice control aspects of its operation.

There are 3 processing chips on the base station controlling each function independently, for: dedicated communication with the Wrote, voice recognition, and completing the main data operations. The communication chip is a Sensium CC981 ultra-low-power wireless enabled sensor interface from Toumaz (TZ1030, with internal module interaction as shown in Figure 3) which contains an embedded 8051 microprocessor and a radio transceiver.

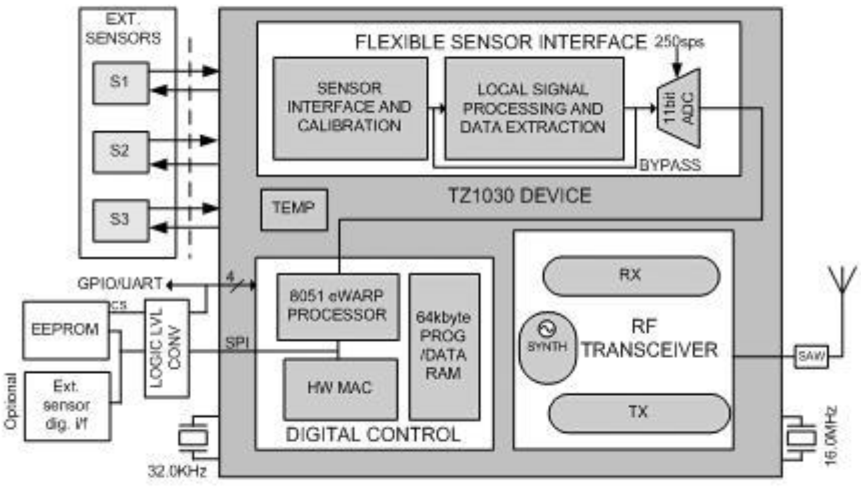


Figure 3 The TZ1030 Ultra Low Power Wireless Sensor Interface

The sensor interface uses AMx™ Mixed Signal Technology [130] developed by Toumaz which exploits the "sub-threshold" region of the transistor, in which CMOS operations consume very little power due to their barely-on state. As a result, the current through such transistors in this region is an extremely low few nanoamps. It was found by Toumaz that during operation in this region, the voltage/current characteristics formed a "well defined exponential" which could be exploited by the system to make use of the physical properties of the transistors as mathematical building blocks for a number of processing functions. The chip is ideal for use in wearable monitoring device applications, as its ultra-low power requirements resulting from real-time processing in the sub threshold, analogue domain mean that the battery size is reduced and lasts much longer than with non-AMx™ technologies. It has a 64kb on-board programming and RAM capability and as such an amount of pre-processing of sensor values such as those obtained for heart rate could occur locally on its 8051 processor, however in the Verity implementation the PIC control chip handles all data operations on the base station and the CC981 acts solely to receive the observation values transmitted from the Wrote by the sister-CC981 located there. Another unique feature of this chip is its built-in temperature sensor which can be utilised to provide a further estimation about the current state of the user's environment when coupled with its partner temperature sensor – again situated on the Wrote. Whilst it also has a number of sensor ports available for attaching subsequent analogue or digital devices to which it can interface, the base station connects its 3D accelerometer directly to the PIC, leaving the CC981 for communication. This secondary accelerometer works to identify the orientation of the user when analysed in correlation with another located on the Wrote.

The chip does have a self-contained radio transceiver and as such the base station has within its package an aerial with which to receive the signals from the wrist device. The aerial connected to the Sensium chip is used solely for communication with the Wrote's sensors, but 2 other aerials on the base station are utilised by the GPS module and the GSM module present for communication via mobile telephony (specifically the Antenova A10340). The GPS module is intended to be available to be called into use during operation in the event that the user either requests their location, or the person in contact with them through the GSM module desires to know where they are currently located in the instance that they are unable to answer for themselves.

The GSM module provides Verity with mobile telephony capabilities once there is also a SIM card connected and registered to the GSM network. On the base station however, there are no interface buttons - nor is there a graphical interface as would be expected of a device incorporating a mobile telephone. Instead of manual input (which may be deemed too complex for the elderly demographic at which Verity is aimed – as discussed in Chapter 2), there is a voice recognition and communication module (from Sensory - the RSC-4128), which communicates directly with the user through natural language and currently accepts simple confirmation responses of Yes, OK and No, as well as a more specific limited vocabulary which is tailored to the user, its general application for the elderly and the speech recognition trees used in the call request or verbal state identification operations, i.e. "*shower*", "*neighbour*", "*daughter*", "*warden*", "*call centre*", "*doctor*", "*emergency services*", "*key holder*" and "*quiet*" are currently all programmed for recognition within the Sensory chip. The voice chip activates when an erroneous event or unusual state transition is detected by the control unit, or when an inbound telephone call is connected. Similar trees activate in the event of a fall and an incoming call, as well as periodically if a sensor input indicates a specific state is occurring e.g. the user may be required to indicate that they are in the "*shower*" as a means of explaining a temperature increase or a lack of detection of any observable signs. All speech trees are given as reference in Appendix B.

The control unit chip of the base station (a Microchip PIC24FJ256GB106) is the primary centre of the data processing, in which all of the intelligent algorithms will be running after capturing the sensor readings. It performs initial recognition of the states of a user and controls the voice-based human-machine interaction when required by the individual processes running the device. A detected event triggers a dialog between the user and Verity mainly for confirmation of a responding action or to help reduce the false-positives of state detection. In this scenario, the user is alerted of a situation by communication (through the speaker) with the device, which is preloaded with a series of statements or questions which relate to a number of scenarios possible during its use. The alert follows a decision tree where at each stage the user is required to either confirm or deny a statement, causing the device to adjust its operation accordingly. The states include observable states and hidden states.

The observable states, such as *Fall*, *On Table*, *No Communication* and *Communication* (the latter two referring to communication between Wrote and base station, again discussed further in Chapter 7) can be determined from sensor and component readings directly, with little to no algorithmic processes requiring a call into use. The typical result of the majority of the *Fall* and *No Communication* states is an emergency dial-out.

Before making a call, the voice chip starts a dialog with the user for confirmation that the detected scenario is indeed correct. Upon confirmation or indeed lack thereof, the base station nominates the most appropriate contact in its list to connect with; the user has the ability to choose whomever they wish to have stored on the device – from relatives to neighbours, or perhaps paid carers and the emergency services. At this point, Verity essentially becomes a mobile telephone with the ability to inform the nominated contact of the user's current observable state and the readings which have resulted in their being involved in the situation. Should the need arise, the GPS module can also activate in order to assist in the locating of the user if they are not present in their own home at the time of an incident occurring, relating to the other call participant the user's coordinates through an SMS text message or speech synthesis.

A selection of hidden states, e.g. *Sleeping*, *Sitting*, *Standing*, *Walking*, *Running*, and *Abnormal* are estimations of the inferable behaviours of a user which are not explicitly determinable from the sensor readings alone like those of a *Fall*. A behaviour classifier is developed in this research for the possible detection of these initially identified states, and as such is the focus of subsequent sections of this work. These detected states can then subsequently be used for lifestyle analysis and abnormality detection.

The Wrote is tasked with gathering the primary data from the user and is designed around the communication chip, as shown in Figure 4. The temperature of both the user and their environment is captured using (respectively) Vernier surface temperature sensors attached to and embedded within the Sensium CC981. A simple piezoresistive pressure sensor (Huake Elec-Tech's HK 2000G), is attached to the device and placed in permanent contact with the radial artery to obtain readings interpretable as pulse values. The 3-axis accelerometer (Murata Electronics Oy's CMA3000-D01) is utilised to provide a value of acceleration experienced by the wearer and also the current orientation of their wrist. The sampled sensor data after pre-processing are stored in an FIFO buffer in the RAM for transmission to the base station. The transmitted data package is defined as in Figure 5.

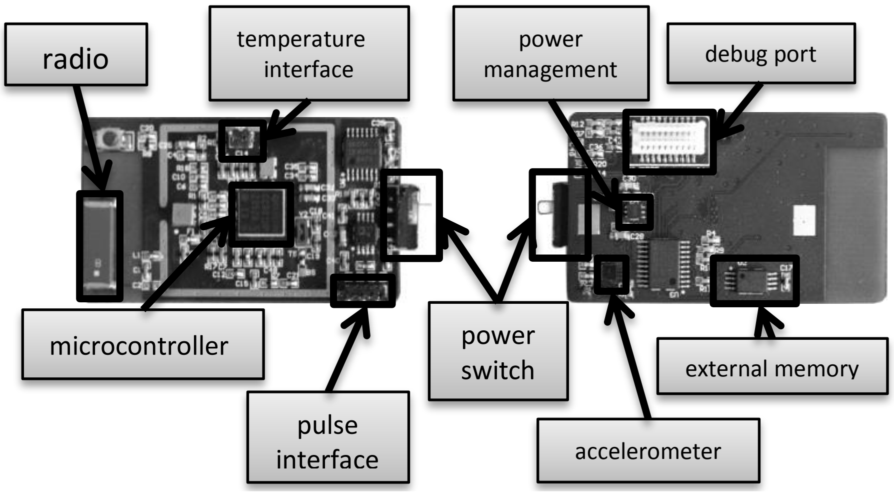
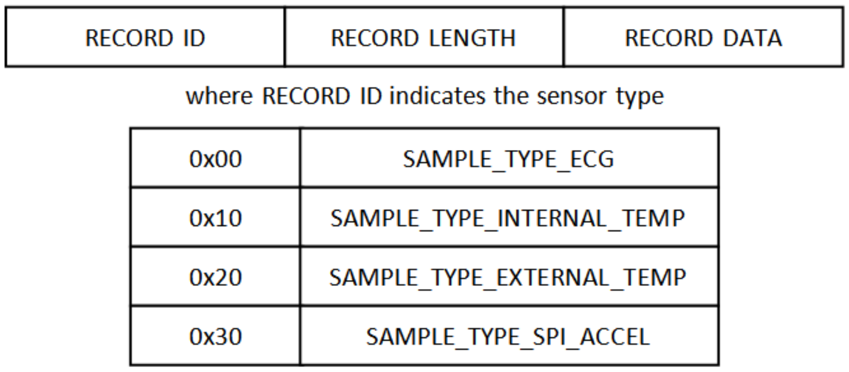


Figure 4 The Wrote layout; the left and right images show the top and underside, respectively



RECORD ID	RECORD LENGTH	RECORD DATA

where RECORD ID indicates the sensor type

0x00	SAMPLE_TYPE_ECG
0x10	SAMPLE_TYPE_INTERNAL_TEMP
0x20	SAMPLE_TYPE_EXTERNAL_TEMP
0x30	SAMPLE_TYPE_SPI_ACCEL

Figure 5 Transmitted data package structure

Communication between the Wrote and the base station is based on the Toumaz Nano Sensor Protocol (NSP) of [131]. The base station first searches for a free RF channel on which to communicate and then allocates it to the Wrote. Once the Wrote is detected, it “handshakes” to establish a communication link for data transmission. The sensor data is then packed into a message and sent to the base station. The base station and the Wrote then go into sleep mode for a time period and wake up ready for the next transmission.

3.2 Sensor Sampling and Pre-processing

Four sensors are incorporated directly into the Wrote for the purpose of detecting a select number of “hidden” states of a user. Skin temperature and heart rate are 2 detectable physiological measurements that are used by Verity as health indicators and to provide early alerts for possible signs of illness. In order to improve the robustness of the detection, ambient temperature, orientation and 3 accelerations are measured as supplementary signals for use in the state decision process, where the 5 currently identified “basic” states can adequately be estimated according to our definitions (Chapter 4). The following sampling and pre-processing steps were devised by the project team alongside the research detailed in this work to provide the behaviour determining models with reliable observation values.

3.2.1 Temperature measurements

The internal temperature sensor of the CC981 chip is used for ambient temperature measurement (that of the user’s surroundings) and an externally connected thermistor is used for skin temperature measurement. The internal temperature sensor is a parasitic PNP device embedded on-chip. With a fixed constant current source, a voltage is generated which fluctuates in response to different experienced temperatures. Through the on-chip 11 bit Analogue-to-Digital Converter (ADC), the voltage is sampled and filtered and further calibrated to reflect the ambient temperature.

A Vernier surface temperature sensor is connected to the chip as an external sensor. With a fast response speed and high accuracy, a 0.03°C (0 to 40°C) resolution and a nominal resistance value of 20KΩ it is more than suitable for the measurement of the user’s skin temperature. This external temperature sensor shares a common interface with the internal one; a constant current source drives the thermistor to generate a voltage which is then sampled through the ADC. The current source is programmable to between 1µA to 12.125µA depending on the requirements.

3.2.2 Acceleration measurement

A 3-axis accelerometer (the same Murata Electronics Oy CMA3000-D01 as on the base station) is used for motion detection. It uses capacitive 3D-MEMS technology and has ultra-low power consumption: 1µA for 1HZ sample rate. The accelerometer is directly connected through the Serial Peripheral Interface (SPI) of the Wrote chip for sending acceleration values to the base station. However, the fall detection operation is implemented on-chip in order to ensure a rapid response. To reduce the amount of computation associated with Euclidean geometry, the L1-norm (Taxicab Norm) is used instead of the L2-norm (the Euclidean Norm) to calculate the intensity of the acceleration vector. The three accelerations from the X, Y and Z axis are therefore combined as absolute values to provide a scalar value for the gravity experienced by the device:

$$\begin{aligned}\text{Acceleration}(g) &= \sum_{i=1}^{n} |x_i| \\ &= |X| + |Y| + |Z|\end{aligned} \tag{9}$$

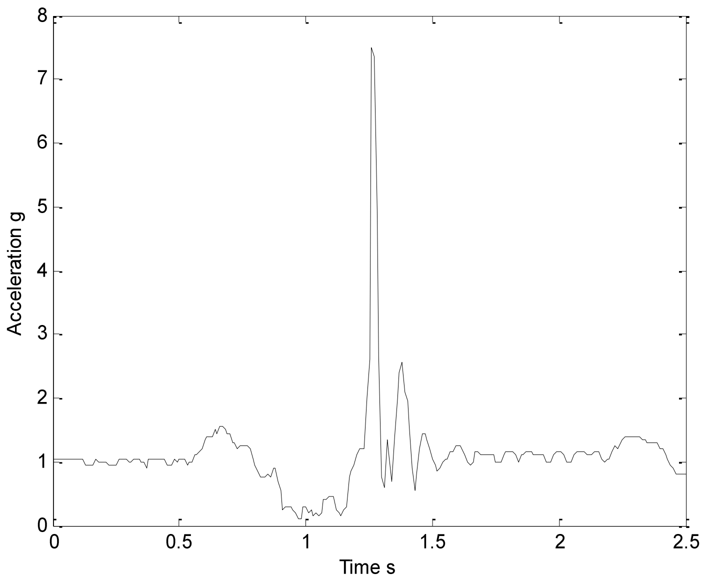


Figure 6 The L1-norm of a fall

The L1-norm describing a fall is shown in Figure 6. It can be seen from the figure that a significant peak of the L1-norm exists for a fall. It can reach 7.5g for about 0.4s and typically a valley of less than 1g occurs before the peak. Based on these observations, a fall detection algorithm was developed:

1. Save every acceleration sample into an FIFO buffer and calculate its L1-norm.
2. If the L1-norm is higher than a predetermined threshold, open a 1 second window for the fall detection operation. A 100Hz sample rate is used, therefore a 1 second window comprises of 50 samples before and 50 samples after the threshold is exceeded.
3. Calculate the L1-norms for the first half window and the last half window. If there are L1-norms of less than a threshold (of less than 1g), a fall is alerted.

3.2.3 Heart rate measurement

Heart rate dynamics has prognostic significance for the progression of several cardiac diseases [132]. Longitudinal analysis of heart rate variability also has the potential to prove clinically useful in differentiating the progression of a disease process, which may be a significant additional functionality of the system. Verity measures heart rate with a piezoresistive sensor (Figure 7), aided by the accelerometer in a bid to “clean” the typically noisy result due to motion obtained by the piezoresistive sensor alone. Simply, the sensor measures the pressure changes at the wrist due to the heartbeat, which in turn affects the resistance of the piezoresistive material.

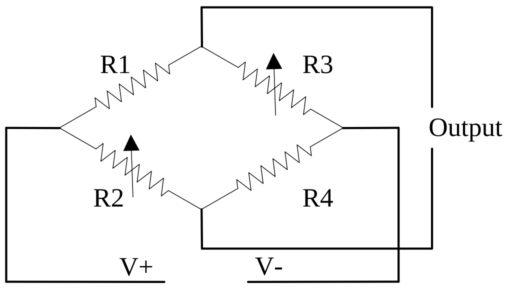


Figure 7 Piezoresistive sensor for gathering pressure data caused by pulse

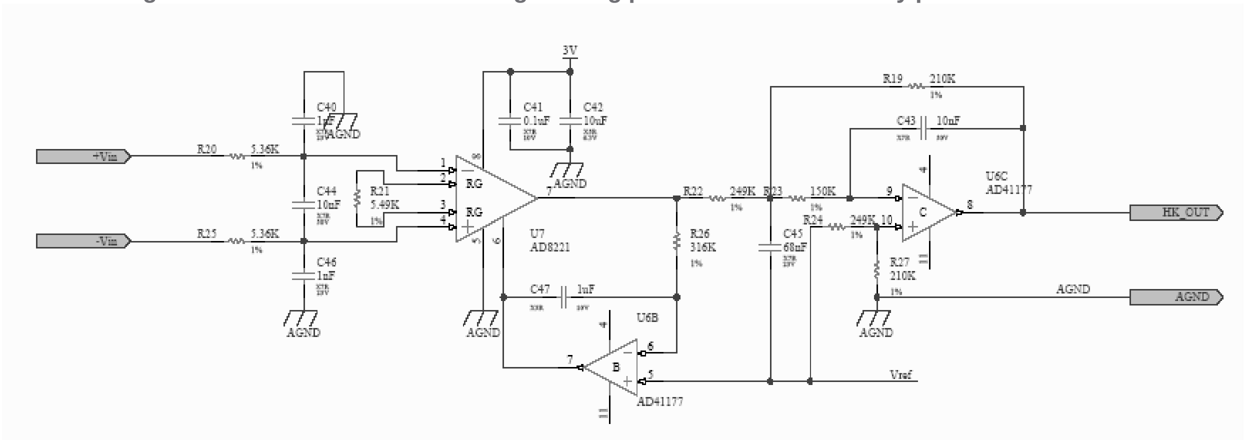


Figure 8 Conditioning circuit for filtering of unclean signals obtained by the piezoresistive sensor

The output voltages of the piezoresistive sensor are sent to a conditioning circuit for the pulse measurement (Figure 8). Because the Wrote is a wireless platform, it inherently suffers from high-frequency interference; the voltages are first filtered by an RC circuit with a common-mode cut-off frequency of 15.9Hz and a differential-mode cut-off frequency of 758Hz. Considering a typical heart rate is measured to between 1 and 2Hz, the differential signal is further fed to a high-pass filter and a second-order low-pass filter. The high-pass filter is designed to have a cut-off frequency of

0.5Hz, with the low-pass filter designed to have a cut-off frequency of 17.6Hz. The output of the conditioning circuit is then sampled by the ADC of the Wrote's control chip for further processing.

The sampled pressure signals are often contaminated by noise mainly due to motion. Two signal processing methods are therefore developed to ensure a more robust heart rate measurement: the first is an adaptive peak detector to overcome signal shift and the second is a Kalman filter [133] which takes into account the motion influence, as described in the literature review of Chapter 2.

The pressure signals are sampled at 250Hz, which, due to being a multiple of 50Hz, may result in them suffering from the work frequency noise. Therefore, 2 cycles of 50Hz – i.e. 10 samples – are averaged to overcome the interference. Due to heart rate variability [132], a heart pulse period can demonstrate a difference of up to 300ms. This causes difficulty for heart rate measurement, especially for restless users if the interference from such motion has a very similar frequency and magnitude to that of a heartbeat. Adaptive peak detection is conducted to extract a heartbeat; the basic idea is to predict the time of the next pulse by setting a blind period during which a detected electric pulse is ignored.

Two thresholds are defined, $vol\_threshold$ and $num\_threshold$, where $vol\_threshold$ is the threshold for a possible peak signal and $num\_threshold$ is the width of the blind period - in this period any peak will be accounted for as interference.

1. Scan n voltages to compare with the $vol\_threshold$. If $v(i) > vol\_threshold$ where $i = 1 \ldots n$, $v(i)$ is detected as a peak.
2. Ignore the next $num\_threshold$ samples, thereby resuming at $i = i + num\_threshold$.
3. Repeat steps 1 and 2 for peak detection of all n samples.
4. Count how many samples between 2 peaks have a pulse period T. Then pulse rate $Z = \frac{1}{T}$ is fed to a Kalman filter for optimal estimation as the value $\hat{x}_k$ used below.

5. Calculate the maximum voltage $v_{\max}$ and the minimum voltage $v_{\min}$ in the n samples.
6. Both thresholds are updated as:

$$vol\_threshold = k_1 \times (v_{\max} + v_{\min}) \tag{10}$$

$$num\_threshold = \frac{k_2}{\hat{x}_k} \tag{11}$$

7. Go to 1 for pulse detection of next n samples using (10) and (11) as the new thresholds.

The proposed peak detection algorithm uses the adaptive voltage threshold (10) to cope with variable pressure strength and the adaptive blind window (11) to cope with the noise from motion. The width of the blind window is adjustable according to the estimated pulse rate given by the Kalman filter.

A Kalman filter is a set of recursive equations which provide optimal smoothing, estimation and prediction of states from sensor inputs: taking into account the system model, its uncertainty and the sensor and noise model. Since its inception it has been a versatile engineering solution that is widely used in signal processing, control engineering, and sensor fusion for such applications as robotic control [50]. Consider a discrete linear system:

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1} \tag{12}$$

where state x and u are system state and input; w is system noise, $p(w) \sim N(0, Q_k)$. The observation function is

$$z_k = Hx_k + v_{k-1} \tag{13}$$

where z is the system output with noise v, $p(v) \sim N(0, R_k)$.

A Kalman filter takes two steps for optimal estimation:

Model based prediction

$$\hat{x}_{\bar{k}} = A\hat{x}_{\bar{k}-1} + Bu_{k-1} \tag{14}$$

$$P_{\bar{k}} = AP_{k-1}A^T + Q_k \tag{15}$$

Observation based correction

$$K_k = P_{\bar{k}} H^T \left(HP_{\bar{k}} H^T + R_k\right)^{-1} \tag{16}$$

$$\hat{x}_k = \hat{x}_{\bar{k}} + K_k \left(z_k - H\hat{x}_{\bar{k}}\right) \tag{17}$$

$$P_k = (I - K_k H) P_{\bar{k}} \tag{18}$$

where K is the Kalman gain. It is adjusted by considering the uncertainty of model P and of observation R.

Van Der Pol [134] stated that heartbeat can be modelled as a relaxation oscillator, whose frequency is dependent on the energy supplied [135]. For a short time window, the energy can be assumed to change less, therefore the pulse rate can be approximated as:

$$x_k = x_{k-1} + w_{k-1} \tag{19}$$

where w is the model uncertainty with a normal distribution $p(w) \sim N(0, Q)$ to represent heart rate variability, which can be caused by several physiological and mental aspects [136 , 137].

The pulse rate x is observed to be z by the piezoresistive sensor in step 4 above, and modelled as:

$$z_k = x_k + v_{k-1} \tag{20}$$

where v is the sensor noise with a normal distribution $p(v) \sim N(0, R)$. From experimentation, the motion of a user can introduce strong elements of noise into the sensor's observations. However, both the amplitude and the time of the motion are not predictable, which makes the extraction of a heart rate reading for "restless" users very difficult. The 3-axis acceleration values obtained from the accelerometer are used as a measure of the motion, which determines the uncertainty of a piezoresistive sensor reading. The variance R_1 for m samples is thus defined as:

$$R_1 = \sum_{i=1}^{m} \left| X_{ACC}(i) - \overline{X}_{ACC} \right| + \sum_{i=1}^{m} \left| Y_{ACC}(i) - \overline{Y}_{ACC} \right| + \sum_{i=1}^{m} \left| Z_{ACC}(i) - \overline{Z}_{ACC} \right| \tag{21}$$

where $X_{ACC}(i)$, $Y_{ACC}(i)$, and $Z_{ACC}(i)$ are accelerations in the 3 directions and $\overline{X}_{ACC}$, $\overline{Y}_{ACC}$, and $\overline{Z}_{ACC}$ are mean accelerations for period m. The absolute modulus is used instead of the mean square root in order to ease computation.

Considering the uncertainty due to motion for the k^{th} sample, the variance of v_k in (20) is defined as:

$$R_k \begin{cases} T_1, R_1 < t_1 \\ T_2, R_1 > t_1 \end{cases} \tag{22}$$

where $T_2 >> T_1$. When $R_1 < t_1$, the user exhibits less motion. Heart rate variability w_k in this case is the main signal present and more emphasis should be placed on the sensor reading z. Therefore, v_k has a lower variance of R_k. When $R_1 > t_1$, there is a stronger instantaneous motion which causes an unreliable sensor reading. The variance R_k is given a higher value to reduce the Kalman gain in (16). The heart rate is then updated by the Kalman filter iteratively through steps (14)-(18), with $A = 1$, $B = 0$, and $H = 1$. Through testing with a preliminary Verity system, this developed motion-adaptive heart rate estimation has been verified to be robust against the motion experienced by a user whilst wearing the device.

Through parameter tuning, a robust heart rate detector is achieved. With hardware filtering and using $n = 1000$, $K_1 = 0.56$, $K_2 = 0.59$ in the peak detection algorithm and $m = 50$, $t_1 = 150$, $Q = 100$, $T_1 = 100$, $T_2 = 850$ in the Kalman filter, this approach can be compared with a mean filter algorithm. An example of the pulse signals and accelerations is shown in Figure 9.

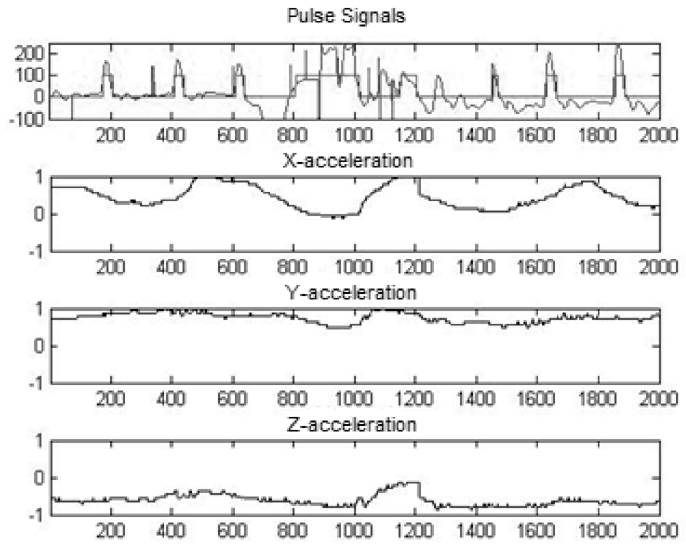


Figure 9 Pulse signals affected by the user's motion

The sampled voltage of the pulse signal can be seen to be the “jagged”, more analogue-appearing trace, with the signal after the peak detection shown as a more digital-appearing trace in the top graph of the figure. The acceleration signals reflect the motion of a user on each axis. For a smoother motion, the proposed peak detection algorithm can effectively overcome noise to extract reliable a heartbeat value. However, when the observed motion becomes more intense, (e.g. 800 to 1000 samples), heartbeats are commingled with motion and are indistinguishable in the piezoresistive signals. The peak detection algorithm alone fails to capture these heartbeats where the Kalman filter approach effectively overcomes interference from motion.

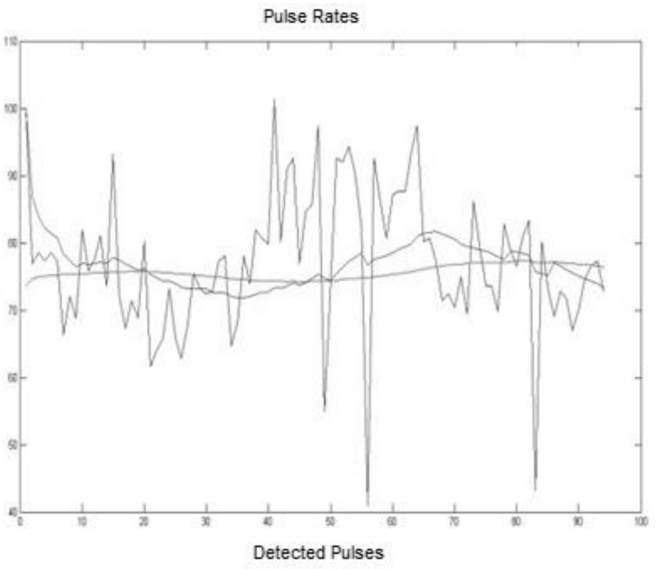


Figure 10 Pulse rate estimation with the Kalman Filter

Figure 10 shows the heart rate estimation with the Kalman filter, in comparison with the mean filtering of the detected pulses. The pulse rate obtained in step 4 is the erratic signal shown, varying between 40 and 100bpm due to the user's motion. The mean of 30 samples is calculated to reduce the influence of motion noise and is shown as the smoother signal, achieving a variance of 12bpm and effectively smoothing out the original signal. The proposed Kalman filter based heart rate estimation is shown as the most linear trace, further exhibiting the fact that the method has cleaned the signal to a more useable value. It is shown that reliable heartbeat estimation is obtained with overall variance of only 2bpm. Therefore, the proposed method effectively deals with the noise from motion for the piezoresistive sensor-based heart rate measurement.

3.3 Sensor Operation in Practice

In order to suitably approximate states of the user based on the information gathered by the sensors, the possible states must be accurately defined in terms of the readings which may be obtained. In such circumstances, testing with a wide variety of subjects would be desirable in order to formulate an average reading for each of the sensors which will typify a state. These averaged readings form the basis of the initial definitions for the predetermined states, with the intention that through use and further operation by a single user the readings typifying states will adapt to be user-specific. An increased number of sensors would result in greater definition for a state, yet with the four sensors available in the system, the fusion will provide relatively certain results regarding the observed state. Ambiguous state assumptions arise when the sensors are accurate yet the state is too similar to another to be distinguished correctly. In this case, the more probable state is determined considering the transition likelihoods.

The sensors used by the system greatly influence the probability of isolating the correct state from the number of likely states. The fusion of sensor readings provides greater certainty of a state belief than with a single reading, yet if the sensors obtaining the readings are unreliable then the fusion result will be falsified. The temperature sensors applied in contact with the body measure the heat emitted at that site. At worst, the sensor will read the temperature of the skin surface as well as the temperature of the containing device – which may vary in temperature given its many constituent components. However in most situations, the sensor will read and return an accurate reading given that it only has the one affecting variable of heat. The same properties apply to the ambient temperature sensor which is measuring the surrounding air temperature, yet the inaccuracies may develop when considering clothing covering the area or affecting the general air flow over the sensor. These two points may further lead to incorrect state identifications if other sensors were not considered.

The accelerometer has a large influence on the state determining as its returned value greatly differs from one predetermined state to the next. Should the two temperature sensors return ambiguous values which may belong to both a state such as *Sleeping* and also to *Walking*, the accelerometer value will indicate which state the user’s motion is typical of.

The sensor most likely to return an inaccurate value is the pulse sensor. Placed over the radial artery close to the wrist, the sensor is intended to detect the pulse through deformations of its piezoelectric membrane. Not many applications regarding heart-rate monitoring with a single sensor (as opposed to multiple electrodes used by ECGs, for example) use piezoelectric membranes and instead opt for the photoplethysmographic approach to detect SpO_2 concentration either in the extremities of the digits or on the wrist. Such devices use a high power due to the constant need to illuminate the light source and monitor the detector. Although this method is subject to some motion influences (distance to the skin can reduce accuracy in detection of the light reflected) it is considered more reliable than the piezo method [138]. Given that the device is low-power and must operate fully with the other sensors however, the piezo method - with its low power consumption characteristics [139 , 140] - was chosen for the pulse detection. The issue arises when deformations are detected which occur not due to the pulse through the artery but to the motion of the user and friction between the membrane and the skin surface. A Fourier Transform of the readings taken by the sensor can provide details of the component frequencies which make up the reading. [140] is a patent for a device which uses the piezo implementation and uses a Fast Fourier Transform to process the data regarding deformations. The observation is that the noise from motion will have less periodic properties than the readings caused by pulse deformations and thus the period of the pulse may still be extracted from the transformed data. In the case of extreme noise which appears to mask the pulse reading, filtering methods may be applied to no avail if the reading itself is not contained in the signal. This is where the other sensors and their influence on the state definition are to be considered with a higher weighting.

3.4 Summary

This chapter has served as an introduction to the Verity system, detailing the construction of the device itself and the initial processes employed to provide the subsequent intelligent methods devised in this work with the most suitable form of data in order to reach a conclusion of the users' behavioural state. The two main separate components of the base station and wrist-mounted device were explained in detail and their internal components' relevance to the behavioural state-determining was explained. The following chapters describe the algorithms and principles developed in the research and utilised in the programming and software implementation of Verity, in order to provide a complete overview of the system as a whole.

Chapter 4
A Hidden Markov Model and Fuzzy Inference System for Behaviour Recognition

Whilst for Verity the observation of a single sensor may have a probability of belonging to many possible behaviour states, a fusion of multiple observations from multiple sensors will provide a more concise selection of states from which they could be emitted. The combination of these observations is of primary concern for this application, as the probability scheme employed within Verity is the Hidden Markov Model (HMM): typically taking as its input a series of singular observations and predicting the most likely state capable of emitting such an observation at each step. The result is a sequence of states which approximate – in this case – the behaviour of the user.

This section details our adaptation of the traditional HMM to incorporate the Fuzzy Inference System (FIS), providing the model with observation probabilities based on the combination of available sensor readings. Also discussed is the initial “intelligent” updating method for the FIS, developed with the intent to fully autonomise the system such that the governing fuzzy rules adapt to the user during operation over time.

4.1 The Adapted Hidden Markov Model

As was discussed in the Literature Review of Chapter 2, the Hidden Markov Model (HMM) is a well-implemented and ideal mathematical model for use within a behaviour monitoring and classification system, which takes observations as inputs and determines the most likely state to emit such observations as a way of estimating the behaviour of a user. Whilst formally identifying the behaviour of a user cannot simply be taken as a mathematical problem due to many influencing factors, modelling behaviour transitions and probabilities of occurrence can provide reasonable estimation of a user's state based on available evidence. For this reason, we employ the HMM within Verity only as a means to estimate behaviour, with subsequent confirmatory checks made using speech-based interaction to further enforce the observation-informed estimation.

The standard HMM is used to model a sequence of discrete, unseen states from a sequence of associated observations and requires an *observation probability matrix,* referred to as *b* in the literature, which details the probability of seeing all possible observations in each of the possible states that can be emitted by the system (in the behaviour monitoring case: the user). For the Verity application, it is difficult to define each observation probability in turn, especially when an observation used in this system currently comprises of more than one reading – and is essentially a multi-dimensional observation vector in itself – due to the sheer number of observations one would have to consider to ensure all possible scenarios are accounted for. The solution in this case is to develop a suitable method to fuse multiple sensor readings into a single state-associated observation probability, bypassing the need to explicitly define or train the probability matrix for each user of Verity.

With the requirement also of Verity to provide subsequent data for a medical professional so that they might be able to diagnose possible health problems with the user, there is some necessity to incorporate natural-language and human knowledge. The belief is that it is better understood the reasoning for a behaviour if it is defined through natural language and has a basis in human knowledge than if the algorithm is programmed in a purely numerical fashion with the HMM obtaining its observation probability values from training data - which could actually be inherently erroneous or ignore vital observation scenarios which inform of a state and were not seen during the training period.

The result is the inclusion of the Fuzzy Inference System (FIS), which directly interfaces with the sensor values as they enter the system to provide the observation probability value in a much more simplistic and human-knowledge-based approach than in the standard implementation: fusing multiple sensor readings into one and determining the observation probability for a state which would typically be found in the standard, discrete probability matrix. Subsequent operations continue as normal, with the FIS considered for all intents and purposes as a separate system within the HMM.

4.2 The Fuzzy Fusion of Sensor Inputs

The application of a Fuzzy System in HMMs is not uncommon. Kelarestaghi, Slimane and Vincent [141] describe an adjusted HMM which modifies all of its algorithms to utilise variations on fuzzy MIN and MAX operators. For example, the usual Forward algorithm's summing and multiplying induction step now takes values determined by the lower probability – that of transition or seeing the observation in a state. Methods of incorporating additive and non-additive fuzzy systems in HMMs are shown by Verma and Hanmandlu [142], using fuzzy re-estimations of the Baum-Welch algorithm for the determining of the HMM parameters. For virtual reality training of Bone Marrow Harvesting, de Moraes and dos Santos Machado [143] discuss a fuzzy approach to return a value of membership of an observation sequence to a state.

The usual observation sequence in an HMM is of the form $O = O_1, O_{2,} O_3 ... O_T$, where O_i will usually be a single random variable to which a probability of occurrence in a state S_i is assigned $b_i(O_k)$. The problem arises when the observation is a combination of many continuous values, such as that of the Verity multi-sensor system. Assigning probability values to each possible combination of sensor readings is a laborious task, after which there needs to be a method of determining the single probability value for that state. Continuous observations have been considered in HMMs previously [36], with a continuous probability distribution replacing the usual observation probability matrix:

$$b_j(x(t)) = P(x(t)|q_t = S_j) \tag{23}$$

In Verity, however, the FIS is therefore solely applied to the $b_j(k)$ values in place of the usual matrix or probability distribution, making it a system independent of - yet incorporated into - the HMM, as in Figure 11 (the grey shaded box is where in the traditional HMM one would find the possible observations and their output probabilities from the states). The parameters of the FIS depend on the number of states in the HMM, as it determines the number of linguistic knowledge rules needed which govern the memberships. Rather than the HMM consulting the observation matrix or distribution, it consults the FIS to return the probability of seeing the combination input sensor values in that state.

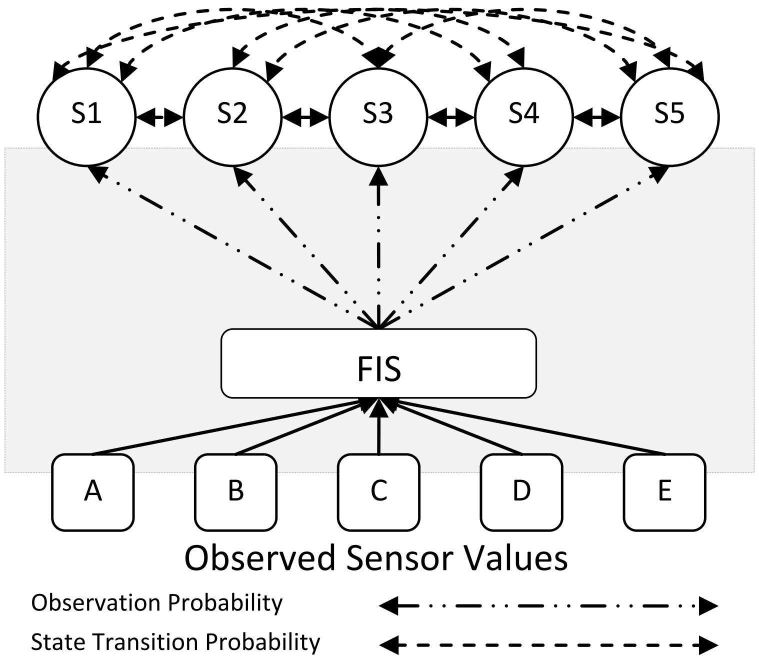


Figure 11 The FIS interaction within the HMM

For each of the Z sensors (λ) making up the observation V there should be G membership functions (f) in which a reading from that sensor can belong. For each of the N states there must be at least 1 rule to determine the activation, given the sensor readings. The number of membership functions within each sensor's range is determined by a combination of human knowledge of the application and required accuracy, but in the general example given below only a small number of memberships are used for simplicity. Firstly the range of readings possible from the sensor must be split into G individual (human determined) fuzzy variables according to the application, then for all memberships within the system the triangular-form fuzzy sets (Figure 12) can be generalised thus (actual memberships are application specific and can be drawn from experience and testing, or relevant data provided by a healthcare professional in the case of this behaviour monitoring):

B_L = lower bound of sensor range

$Q_{(i)C}$ = key value of individual range (median), $i = 1,\ldots,G$

B_U = upper bound of sensor range

$$f(1) = \int_{B_L}^{Q_{(1)C}} (1|x) + \int_{Q_{(1)C}}^{Q_{(2)C}} \left(\frac{Q_{(2)C} - x}{Q_{(2)C} - Q_{(1)C}} |x \right) \tag{24}$$

$$f(i) = \int_{Q_{(i-1)C}}^{Q_{(i)C}} \left(\frac{x - Q_{(i-1)C}}{Q_{(i)C} - Q_{(i-1)C}} |x \right) + \int_{Q_{(i)C}}^{Q_{(i+1)C}} \left(\frac{Q_{(i+1)C} - x}{Q_{(i+1)C} - Q_{(i)C}} |x \right) \qquad 1 \le i \le (G-1) \tag{25}$$

$$f(G) = \int_{Q_{(G-1)C}}^{Q_{(G)C}} \left(\frac{x - Q_{(G-1)C}}{Q_{(G)C} - Q_{(G-1)C}} |x \right) + \int_{Q_{(G)C}}^{B_U} (1|x) \tag{26}$$

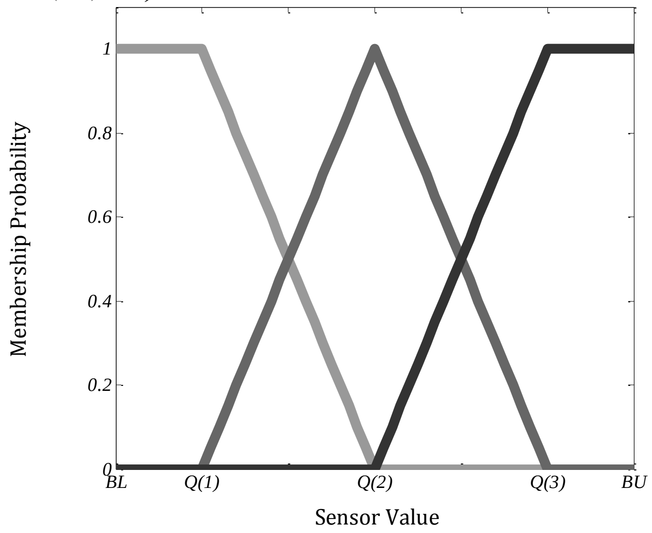


Figure 12 An example of 3 membership functions for a single sensor, using the generalising equations and the Median "key" value

For each individual sensor this will give *G* membership functions which can be descriptively tagged with names such as "*low*", "*medium*" and "*high*". The advantage of this method over discretizing every possible observation output to a probability is that for a fuzzy definition, only a single sensor reading which typifies the membership $Q_{(i)C}$ needs to be known for it to become the centre point of the function; for example a membership of "hot" body contact temperature may be centred on 38° but as the system is used increasingly, the membership may be seen to be more relevant when centred at 37°. This inclusion of an element of human reasoning in the system gives a greater degree of accuracy in the estimation of observation probabilities, as the state definitions are themselves based on human knowledge. With the shape of the membership function in place, the key values describing the membership (i.e. its median) can easily be updated as the system is used.

The rules which activate each state in the FIS are ideally constructed from linguistic rules provided by a physician - or knowledgeable third-party - for each user. It is reasoned that they will have a greater idea of what readings each user should exhibit whilst in the states defined in the model, and therefore the determined probabilities will be more appropriate as they consider this element of human knowledge. In this way, they may also be able to view the data obtained by the system and have a greater understanding of the user's condition at the time. In the first instance a general rule base for all users can be formed at the time of programming the system, which will - as usage increases - adapt to the user according to the feedback received in operation. A fuzzy rule is constructed from the membership functions for each sensor, for example with 4 sensors, where (λ_z) is the zth sensor reading:

"State 1 is activated if λ_1 *is* LOW*,* λ_2 *is* LOW*,* λ_3 *is* HIGH *and* λ_4 *is* MEDIUM"

Once the fuzzy rules are linguistically defined for each state (S_j), its degree of activation $\mu\left(S_j\right)$ in the FIS is considered using Mamdani's min-operation method [144] over all sensor activations, $\mu(\lambda_{1\le z\le Z})$.

$$\mu\left(S_j\right)=\mu(\lambda_1)\wedge\mu(\lambda_2)\wedge\ldots\wedge\mu(\lambda_z) \qquad 1\le z\le Z \qquad (27)$$

With Z sensors, a maximum of Z conditions make up a rule for one state observation probability value $b_j(k)$. Therefore the probability of all sensors producing the corresponding observation value O_i in that state, expressed in HMM terminology:

$$b_j(k)=\min_{1\le z\le Z}\left[P\left(O_{kz}\left|\mu(\lambda_z),q_t=S_j\right.\right)\right] \qquad \begin{array}{r}1\le j\le N\\ 1\le t\le T\\ 1\le k\le M\end{array} \qquad (28)$$

Updating the entire HMM with a standard method such as the Baum-Welch algorithm [26] is inappropriate if the observation probabilities are obtained from a fusion of multiple sensors governed by a fuzzy system of human reasoning rather than being determined through training with example sets, as activation of a state and the sensor fusion weighting is subjective according to the user and the knowledgeable professional. The transition and starting probabilities for the states, however, can continue to be updated using the Baum-Welch approach if the FIS updates with a more tailored method of learning from examples which expand on the already-known relationships. During operation it may be that a new observation with an undefined state membership occurs, which would require the triggering of a confirmation scenario with the user or professional as a means of identifying the current state, so that its activation conditions can be included in the rule base. A rule-learning method such as that proposed by Hong and Lee [145] suffices to take a series of multi-dimensional observations and their corresponding state activations as a supervision signal in order to infer a relationship between the inputs and outputs, producing linguistic rules which describe the input conditions required for each state activation. The scheme is used in Verity’s rule induction step with the addition of a best-approach measure developed herein to enable the creation of the most concise rule base from given data.

4.3 Induction of the Fuzzy System Rules

This technique takes as its inspiration that put forward by Hong and Lee [145], modified and with an additional decision measure which assesses the initial database of input observations to ascertain the best dimension to begin the rule learning with, to ensure that the most concise rule base can be formed for a given FIS consisting of more than 2 dimensions. The original method put forward by the authors used a clustering technique to group the output states according to similarity due to their continuous-value nature, returning a result after fuzzification, rule inference and defuzzification which required hard-limiting to ascertain the state an observation belonged to. Here the technique is modified to deal with - and output - probabilities of occurrence of discrete states, assuming P inputs (dimensions, d , of the input data) are required to identify a state - indicating the presence of P antecedents per rule. The membership functions are assumed to be triangular as described above, with the median value governing the central peak of a function and therefore the lower weighting of contribution an observation has as its value tends to the limits of the triangular function.

The aim is to build a rule matrix which consists solely of activated cells, from a rule matrix which is initially sparse due to the presence of observations in all dimensions which together don’t activate a state (Figure 13).

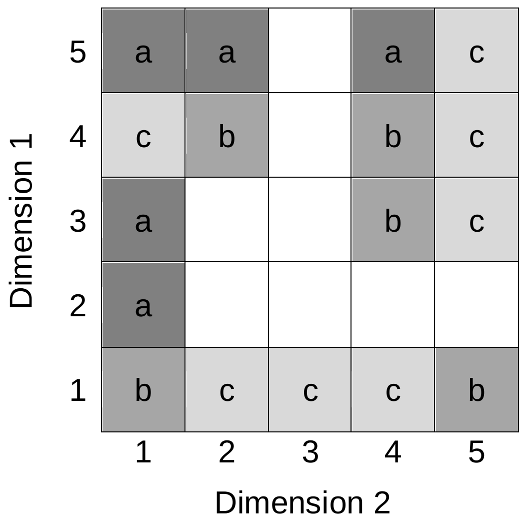


Figure 13 Example rule matrix for a *P*=2 dimensional input fuzzy system, with 3 state outputs

An activated cell indicates the presence of a state in all P dimensions at that location, for example in the unprocessed rule matrix of Figure 13:

a $=\left[\left(2\le input_1\le 3\right)\vee\left(input_1=5\right)\right]\wedge\left[\left(1\le input_2\le 2\right)\right]$ and

a $=\left[\left(input_1=5\right)\right]\wedge\left[\left(1\le input_2\le 2\right)\vee\left(input_2=4\right)\right]$

What can be seen in this 2 dimensional example is that gaps in activated cells for state “*a*” exist which could be closed, thus removing the need for an OR ($\vee$) operation and thus removing one of the antecedents of the above rules. Over a large input space this can eliminate a considerable number of rules and antecedents from the rule base, which when more observations are added would be significantly large if there was no method employed to eliminate the unnecessary activation conditions. This procedure reduces the time required to parse a rule base and allows for inclusion of other observations not seen or predicted by the original human knowledge governing the observation probability determining procedure.

Step 1: Identify Key Values

The smallest difference between values in each dimension (x) is first required to be found in order to determine their resolution in the rule matrix (for example, in Figure 13 the resolution for each dimension is 1 due to the discrete integer values which make up an input). This step is akin to a clustering procedure in Instance Based Learning (IBL), sorting the data (29)-(30) to identify the relationship between adjacent values through a *k*-Nearest Neighbours approach. Here though, as we deal with the sorting of each dimension in turn, the Euclidean distance metric is essentially a standard subtraction (31) between adjacent values in a dimension, with the smallest difference providing us with the value of a $unit$ (32). The $unit$ value informs of $number$, which is the number and initial support of all membership functions belonging to that dimension, and the size (resolution) of each dimension in the multidimensional matrix (33).

$$unsorted_x = [x_1, x_2, x_3, \ldots x_n] \tag{29}$$

$$sorted_x = \mathrm{sort}[unsorted_x] \tag{30}$$

($sorted_x$ is then re-indexed for the next step)

$$difference_x = [x_{2-1}, x_{3-2}, \ldots x_{n-(n-1)}] \qquad \begin{array}{r} 1 \le x \le P \\ n = \text{number of input valu es} \end{array} \tag{31}$$

$$unit_x = \min[difference_x] \tag{32}$$

$$number_x = \left(\frac{\max[sorted_x] - \min[sorted_x]}{unit_x} \right) + 1 \qquad 1 \le x \le P \tag{33}$$

Step 2: Memberships

The initial membership functions for each dimension are all triangular with $2 \times unit_x$ support (Figure 14). For ease of simplification during the merge, each dimension has an array for the start (a), triangular peak (b) and finish (c) of each membership, much like the Q_1, Q_2 and Q_3 of Figure 12. Each dimension range starts at 0 and is symmetrical about the centre of the range of known inputs.

Each dimension of the data has $number$ membership functions, each triangular and described by the following equations.

$$b_x(1) = \min\left[sorted_x\right] \tag{34}$$

$$b_x(number_x) = \max\left[sorted_x\right] \tag{35}$$

$$b_x(i) = b_x(i-1) + unit_x \tag{36}$$

$$c_x(1) = b_x(1) + unit_x \tag{37}$$

$$c_x(number_x) = \max\left[sorted_x\right] + \min\left[sorted_x\right] \tag{38}$$

$$c_x(i) = b_x(i+1) \qquad \begin{array}{r} 2 \le i \le (number_x - 1) \\ 1 \le x \le P \end{array} \tag{39}$$

$$a_x(1) = 0 \tag{40}$$

$$a_x(i) = b_x(i-1) \qquad \begin{array}{r} 2 \le i \le (number_x) \\ 1 \le x \le P \end{array} \tag{41}$$

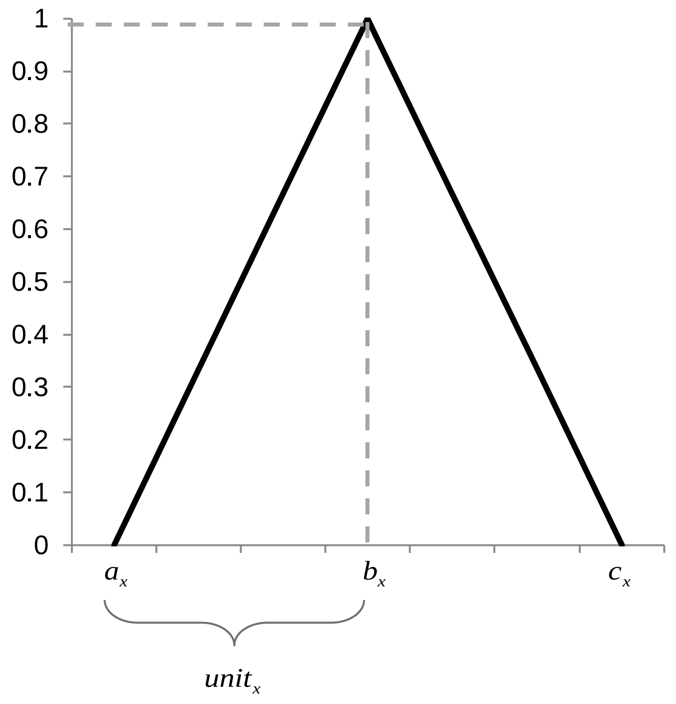


Figure 14 Basic triangular membership function for the fuzzy rule induction

Step 3: Rule Matrix

A combination of antecedents activates a state, as with any FIS. The values of the antecedents are determined in relation to the cells of the rule matrix and the location of the common cell holds the value of the activated state (i.e. $\left[d_1, d_2, d_3, \ldots, d_p\right] = state$) with an assumed maximum fuzzy membership value. If the input value falls within the bounds (the support, in membership function terms) of the cell, it is known to belong to that cell. T is the total number of observations in the set. Figure 15 shows the rule matrix for a 2 dimensional data set and the associated values according to the above definitions.

$$cell_x(i) = j \qquad \{j \in \mathrm{R} : (unsorted_x(i) \geq a_x(j)) \wedge (unsorted_x(i) \leq c_x(j))\} \qquad \begin{array}{c} 1 \leq i \leq T \\ 1 \leq x \leq P \end{array} \tag{42}$$

$$RuleMatrix[cell_x(i), cell_{x+1}(i), cell_{x+2}(i), \ldots, cell_P(i),] = state \qquad \begin{array}{c} 1 \leq i \leq T \\ 1 \leq x \leq P \end{array} \tag{43}$$

The total size of the rule matrix, and therefore populatable number of cells, is equal to the total number of membership functions over all dimensions i.e. $number_x \times number_{x+1} \times \ldots \times number_P$.

	$cell_1(5)$	S_1	S_1		S_1	S_3
	$cell_1(4)$	S_3	S_2		S_2	S_3
$number_1 = 5$	$cell_1(3)$	S_1			S_2	S_3
	$cell_1(2)$	S_1				
	$cell_1(1)$	S_2	S_3	S_3	S_3	S_2
		$cell_2(1)$	$cell_2(2)$	$cell_2(3)$	$cell_2(4)$	$cell_2(5)$
				$number_2 = 5$		

Figure 15 Example rule matrix now formatted ready for the merge operations

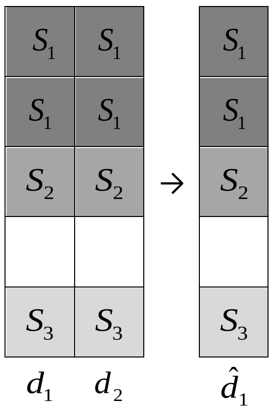
Step 4: Merging Operations

A review of the literature returns a number of techniques created for the condensing and re-evaluation of a rule base in a fuzzy system. For example, that proposed by Nefti, Oussalah and Kaymak [146] operates on the membership functions directly, combining those adjacent to each other with contributing data to the same rules in order to form a single function. Functions with a higher population of contributing data weight the resultant single function to have a higher value about the original's peak. Rule induction is performed on the new functions, with explicit peak values forming the basis of a rule – however multiple instances of the same rule can occur for functions thus requiring further condensing of the rule base to eliminate reproductions. There are 5 merging operations outlined in the original paper of Hong and Lee, intended to reduce the rule matrix to the point where the occupied cells constitute more of the matrix than the empty cells to ensure the most concise set of linguistic rules possible is formed and therefore identify this technique to be most appropriate to apply to Verity. The operations work on adjacent rows and columns of cells of each dimension of the data in the matrix, where here the examples show 2 dimensions for ease of explanation and visualisation. A row or column describes the range of a membership function for one dimension of the data, with any state recorded in a cell of that row or column indicating that it can be found within the bounds of that dimension's fuzzy range.

Each time a row/column is merged in the matrix to eliminate empty cells or combine those that contain the same state, (i.e. $d_x \wedge d_{x+1} \rightarrow \hat{d}_x$) , the membership functions are also updated by amending the parameters in the order as written below. In this example, the term "[ ]" signifies "empty list" i.e. remove an entry; and for all updating processes, $2 \leq i \leq number_x$, with $x = 1$ here.

Circumstance 1:

If all cells in adjacent rows/columns are the same.



Update Memberships**:**

The second of the two identical rows/columns is removed; the membership function start value and the previous membership function end and central values are removed to create one large membership function, incorporating both ranges into one.

$$a_x(i) = [\ ] \tag{44}$$

$$c_x(i-1) = [\ ] \tag{45}$$

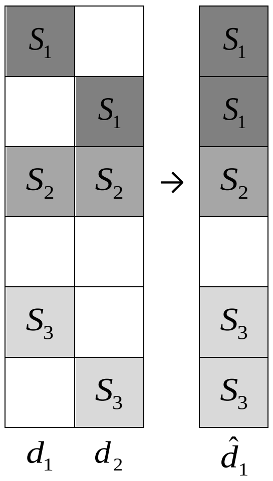
$$b_x(i) = \left[\frac{b_x(i) + b_x(i-1)}{2} \right] \tag{46}$$

$$b_x(i-1) = [\ ] \tag{47}$$

$$RuleMatrix[i,:,:,:] = [\ ] \tag{48}$$

Circumstance 2:

If adjacent cells are the same or one is empty, with at least one cell in the row/column not empty.

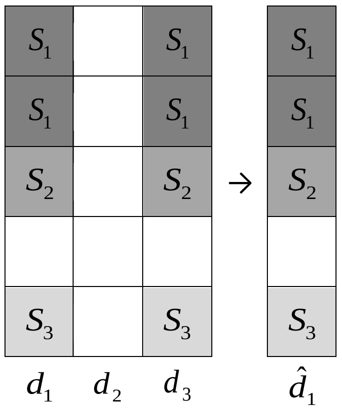


Update Memberships:

The update process is the same as Circumstance 1, removing one of the rows/columns and incorporating two membership functions into one.

Circumstance 3:

If an entire row/column is empty and the cells in adjacent rows/columns are the same.



Update Memberships:

The middle empty row/column is removed; the second of the identical rows/columns' membership function start value and the first membership function end and central values are removed to create one large membership function, incorporating the range spanning the empty row/column into one.

$$a_x(i+1) = [\ ] \tag{49}$$

Equation (44)

$$c_x(i) = [\ ] \tag{50}$$

Equation (45)

$$b_x(i-1) = \left[\frac{b_x(i) + b_x(i+1) + b_x(i-1)}{3}\right] \tag{51}$$

$$b_x(i+1) = [\ ] \tag{52}$$

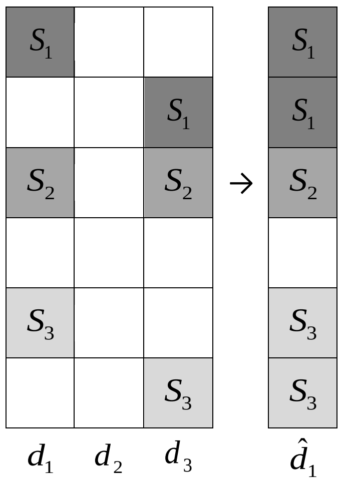
$$b_x(i) = [\ ] \tag{53}$$

$$RuleMatrix[i+1,:,:,:] = [\ ] \tag{54}$$

Equation (48)

Circumstance 4:

If an entire row/column is empty and the cells in adjacent rows/columns are the same or either is empty.

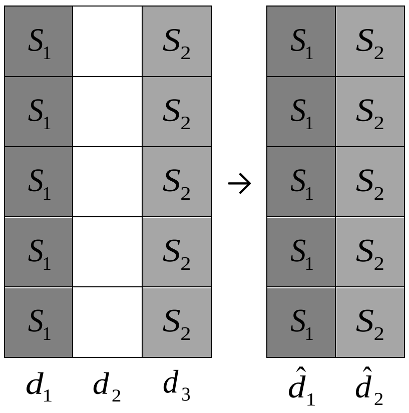


Update Memberships:

The update process is the same as Circumstance 3, removing the empty row/column and combining the adjacent two to populate all possible cells in the remaining row/column.

Circumstance 5:

If an entire row/column is empty and all the adjacent cells on one side have the same region, and all the adjacent cells on the other side have the same region yet one different from the other, merge the three into two.



Update Memberships:

The middle empty row/column is removed; the second row/column membership function start and central values and the first row/column membership function end and central values are removed. The end value of the first row/column function and the start value of the second row/column function take the central value of the empty row/column, to create two larger membership functions where there were originally three spanning an empty row/column.

Equations (49),(45),(53),(48).

Step 5: Rule and Membership Function Creation

The merge steps created a condensed rule matrix which can now be interpreted to give the rules. After the merging, each cell $[d_1, d_2, d_3, \ldots, d_p] = state$ is used in the formulation of a single linguistic rule:

$$\text{IF}\left(input_x = d_x, input_{x+1} = d_{x+1}, \ldots, input_P = d_P\right)\text{THEN } output = state$$

The membership functions are also reduced solely to those that activate the rules. A typical triangular function now has a start-point of $a_x(i)$, peak at $b_x(i)$ and end-point of $c_x(i)$.

It can be seen that for every non-zero cell in the matrix, a rule can be formed. In some cases, reviewing the rules and the associated membership functions will lead to the discovery that some dimensions of the input data will have no bearing on the output i.e. a dimension may only have a single membership function and therefore all values observed from that dimension fall into the same function. Due to the nature of the merging operations these insignificant dimensions' functions will be a constant 1 over the range of inputs when viewed graphically.

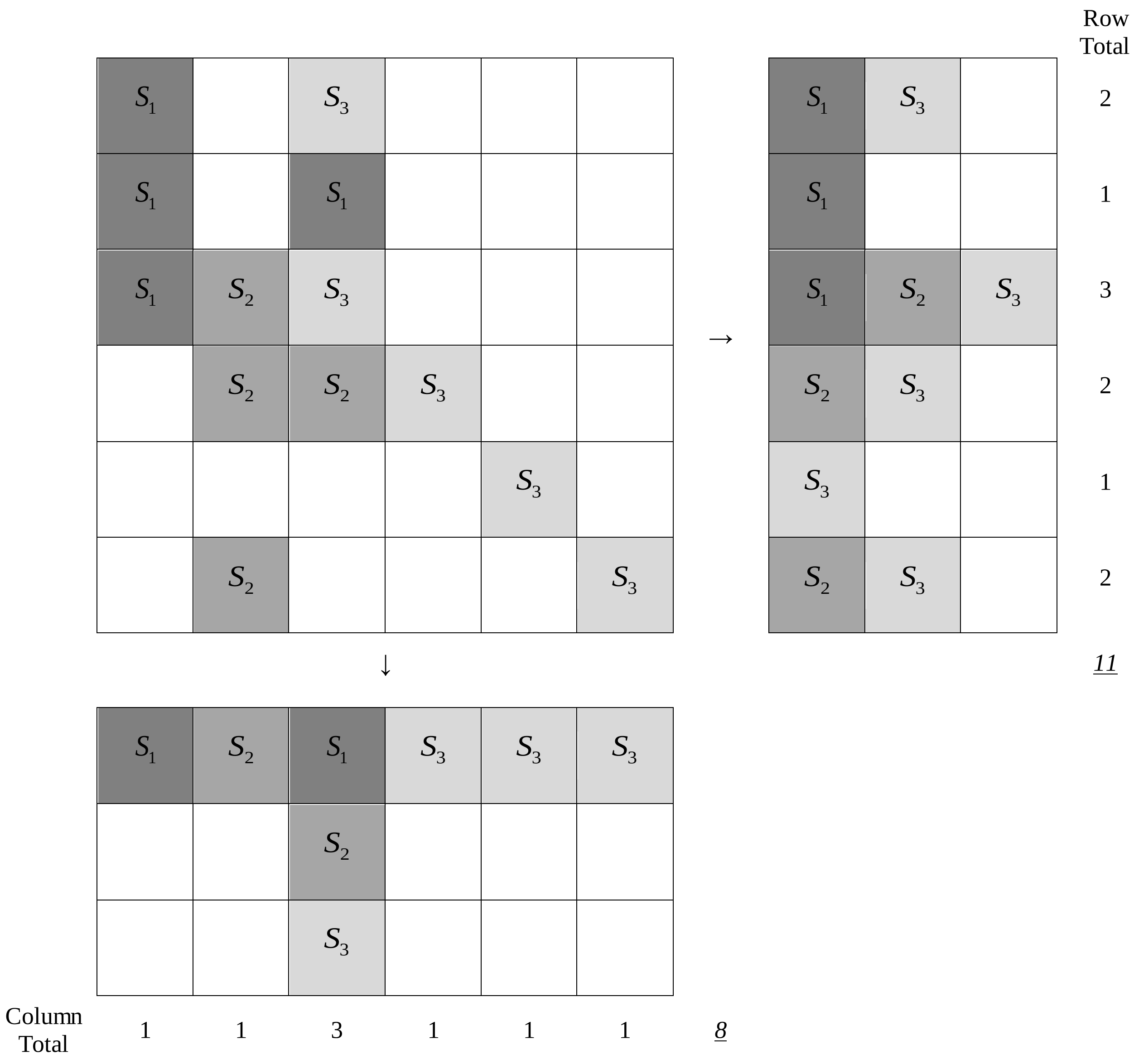


Figure 16 Determining best merge start axis by assessing the number of states in each dimension

The 5 merge operations are carried out in the order specified here so that the best merge can be achieved. It is possible however to have some variation in the number of rules identified depending on which order the axes are addressed in each merging operation. Following Hong and Li [145], the assumed default merge process is 1st dimension, 2nd dimension etc. for each operation. However, through a visual inspection of 2D data, it can be seen that merging first in one axis will result in a smaller rule base than merging in another. Thus, determining the best axis in which to begin merging can increase the efficiency of the process and therefore the efficiency of the rules. For the purposes of reference in this work, we term the merging decision condition rule to be the “Unique Measure Merge”.

As shown in the initial matrix of Figure 16 - created ready for the merge operations - if we were to compute the number of unique entries belonging to a single row/column and sum the total unique entries per axis, we can see that if one axis has a smaller total then the merging as per the above operations will result in a more concise table, and therefore less rules. It may be the case that the number of total rules is less than this preliminary total; with the above example the lower column merge shows 8 unique active cells yet the merge operation will condense the 3 S_3 states on the right of the matrix into a single rule and the row merge shows 11 unique active cells yet the S_1 states present at the top will condense into a single rule – resulting in rule bases of 6 and 10 respectively. Applying this theory throughout the merge operations in each dimension should result in a more general rule base for the given data and thus a more concise set of rules.

4.4 Preliminary Tests

With the models to be used for the behaviour detection identified and now modified to be fit for purpose within the Verity system, a test platform was developed to ensure that the correct approach was being taken to solving the behaviour monitoring task.

The ideal solution to creating a simple verification interface was through use of MathWorks' Matlab software and creating *theoretical* outputs which would be produced by the Verity device during actual operation. In order to design the interface accurately and appropriately, now after having identified the possible mechanisms to be implemented, the parameters required determining.

Through collaboration with the project's partner, iMonSys, as a result of their discussion with an advising General Practitioner: a selection of primary states was identified with which to prove the concept was correct. Taking as an example an average person belonging to the "elderly population", the daily tasks one might face were broken down into simplistic descriptive actions, as per examples in Table I. It can be viewed that the base states are extreme abstractions of the activity, however physiologically the observations on the body would be deemed to be considerably similar on average. It may be more appropriate to assign each state based on exertion levels, in which case a range of 1-5 would indicate high-intensity strain at the upper end, with the lower end indicating a period of rest. However, for the sake of ease of explanation to the user and visualisation of the data, the base states remain as the verb labels in Verity and throughout this work; states 1-5 are, respectively: *Sleeping*, *Sitting*, *Standing*, *Walking* and *Running*.

Table I Examples of how an activity may be classed as a descriptive state

Activity	***Base State***
sleeping	Sleeping
lying down	Sleeping
sitting	Sitting
watching television	Sitting
reading	Sitting
working at a computer	Sitting
eating and drinking	Sitting
sitting	Sitting
bathing	Sitting
standing	Standing
showering	Standing
washing up	Standing
ironing	Standing
cooking	Standing
walking	Walking
shopping	Walking
cleaning	Walking/Running
gardening	Walking/Running
running	Running
vacuuming	Running

These 5 states are certainly not exhaustive in describing human behaviour; as was discussed in the literature review states can be interleaving and be comprised of multiple observable states when examined. In an example of *Golfing*, the user of the system is observable to be perhaps *Running* as well as *Standing* according to Verity, who is observing the user through the device situated on the wrist and receiving readings of high motion as the arm is swung, yet the base station situated in the pocket is reading a less significant motion indicator and thus causing a disparity between states. Here we therefore make the distinction that Verity is capable of adequately identifying instances as defined in Table I, where the placement of the system on both the wrist and in the pocket affords readings typical of those states with respect to motion and other observable signs coming from the sensors. Over time the perhaps infinite number of determinable states will be accounted for with an expanded Verity system, however in this implementation we focus on identification of only these select few.

The 5 current hidden states of the HMM in the Verity application are initialised - with transition probabilities between each defined based on expectation and reasoning as explained above with the assistance of a knowledgeable professional. Operation of the system may later inform of better transition likelihoods, which through an intelligent updating method (such as the typical Baum-Welch approach with expectation minimisation) can replace those put forth initially. However, to remain as a system operating under human assumptions and knowledge, the foreseeable approach is as outlined here. The starting probabilities are contained within a transition matrix of size determined by the number of possible states within the system. In this initial case, transition probability a_{ij} is given by a matrix of 5-by-5 incorporating the above named states:

$$a_{ij} = \begin{bmatrix} 0.45 & 0.35 & 0.20 & 0 & 0 \\ 0.25 & 0.35 & 0.30 & 0.10 & 0 \\ 0 & 0.35 & 0.20 & 0.35 & 0.10 \\ 0 & 0.10 & 0.25 & 0.40 & 0.25 \\ 0 & 0.10 & 0.15 & 0.25 & 0.50 \end{bmatrix} \tag{55}$$

The transition from one state through to the next according to the above matrix is therefore:

1. *Sleeping* to *Sitting:*0.35.
2. *Sitting* to *Standing*: 0.30.
3. *Standing* to *Walking*:0.35.
4. *Walking* to *Running*:0.25.

With the exception of *Standing*, it can be seen that in the assumptions of the above transition probabilities each state has a higher probability of remaining the current state than transitioning out of it – therefore requiring an observation change with a significantly large enough probability to cause a state change within the HMM. Quite often the state of *Standing* was seen to not be a terminal state in any sequence, with a transition to *Sitting* or *Walking* more likely – hence the higher probabilities of transition for both of these states from here.

In all subsequent operations within Verity, the transitions are effectively defined according to an increasing level of exertion, from *Sleeping* as the first state through to *Running* as the fifth. The principle of transition and "allowed" transitions can be summarised in Figure 17.

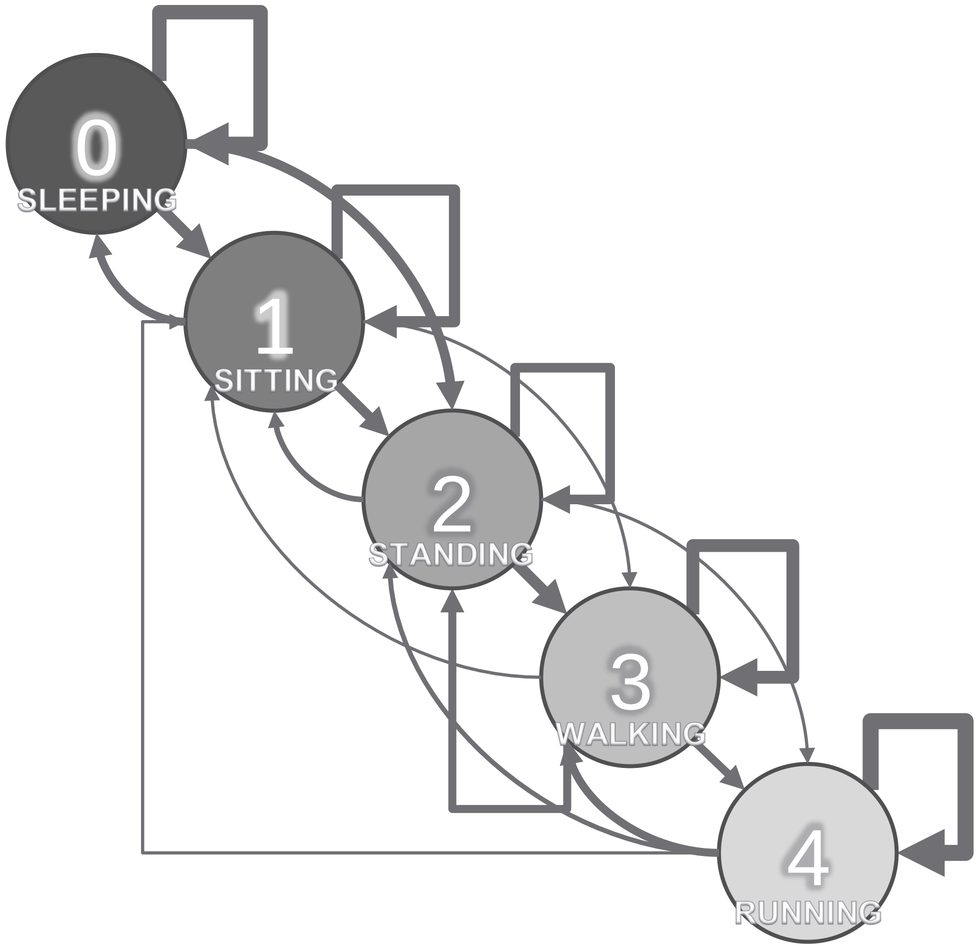


Figure 17 Initial state transitions within Verity, with line weight between states denoting probability strength

Once the sensors' fuzzy membership probabilities are identified, the linguistic rules which activate the states can be constructed based on prior knowledge and assumptions regarding the conditions in which each are observed:

0. **Sleeping**: Ambient Temperature is *Hot*, Contact Temperature is *not Cold*, Pulse Reading is *Normal* and Acceleration is *Nil*

   It is assumed that a sleeping user will be in a warm bed, with a contact temperature varying between normal and hot based on typical sleeping conditions. The pulse rate should be relaxed, and for the majority of the sleep period, average acceleration should be zero.

1. **Sitting**: Ambient Temperature is *Normal*, Contact Temperature is *Normal*, Pulse Reading is *Normal* and Acceleration is *Nil*

   Sitting typically occurs in a comfortable environment, providing the assumption that both the environmental and contact temperatures will be considered normal, along with the pulse reading remaining normal due to no exertion – informing of the zero acceleration value also.
2. **Standing**: Ambient Temperature is *not Hot*, Contact Temperature is *Normal*, Pulse Reading is *Normal* and Acceleration is *Nil*

   Given that standing is in the test case seen as an "intermediate" state, with a transition into sitting or walking usually imminent, the user is assumed to be in an outside or transient environment and therefore could vary between room and outdoor temperatures. The contact temperature would be normal, along with the no-implied-exertion normal pulse reading and zero acceleration.
3. **Walking**: Ambient Temperature is *not Hot*, Contact Temperature is *Normal*, Pulse Reading is *not Low* and Acceleration is *Minimal*

   Again walking assumes space available to move in, therefore a cool or moderate environmental temperature with a normal body temperature. Depending on the user, the pulse can vary from a rest rate to a more motion-indicating rate and is therefore not likely to be low; the acceleration here is upgraded to minimal which is dependent on the speed of the user.
4. **Running**: Ambient Temperature is *not Hot*, Contact Temperature is *not Cold*, Pulse Reading is *High* and Acceleration is *High*

   Running provides the most detectable state from the observations, as the extremes of the ranges are most likely to be observed whilst a user is in motion. Running (or heavy exertion) would imply a comfortable environment and therefore wouldn't be too warm, and the temperature of the user should not be cold as they exert heat during motion. The pulse would understandably be higher than in any other state, with the acceleration following the same reasoning.

These state definitions are currently "basic", with the expectation and understanding that with use and experimentation, more states may be required to describe more situations and that the above definitions may be amended. Distinction of the activities as identified in Table I require testing with the device to ascertain motion levels and temperature/pulse changes throughout the actions, with the assumption that different users will exhibit varying physical changes not generalisable as with the 5 base states due to differing behavioural habits and personal traits i.e. motor/coordination issues, the user's weight or even left handedness influencing an eating style etc.

The basic assumptive rules result in the fuzzy memberships for each of the sensor readings describing shapes as in Figure 18. With all variables now defined either explicitly or as functions describing ranges (the memberships), the test sequence of observations for our 5 possible states (Table II) is loaded to the model such that the system operates as in real-time in the simulated environment of Matlab: taking one observation vector as an input with which to determine the possible state that produced it. Both the Forward-Backward and Viterbi sequence determining methods (Appendix A) were employed to assess their effectiveness, providing quantitative information on their suitability for the behaviour monitoring.

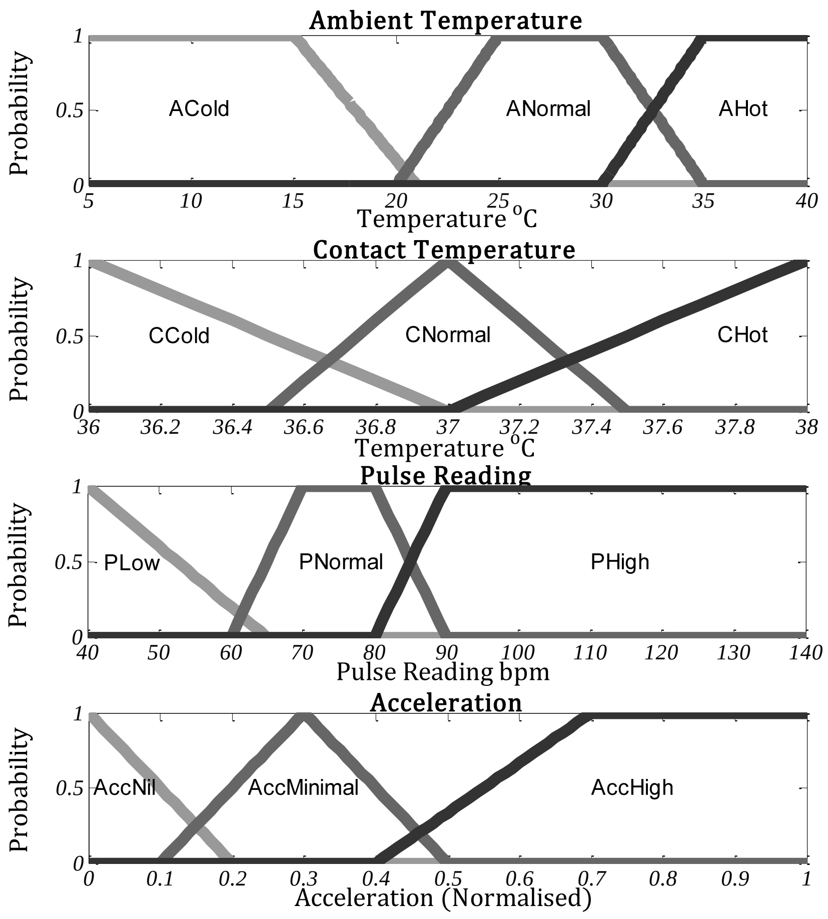


Figure 18 Preliminary test system membership functions

Table II Observation sequence used for testing of the HMM with actual and detected states

Observation	*Temperature*				*State*		
	Ambient	*Contact*	*Pulse*	*Acceleration*	*Actual*	*Viterbi*	*Forward-Backward*
1	35	37	75	0.01	Sleeping	Sleeping	Sleeping
2	35	37	75	0.01	Sleeping	Sleeping	Sleeping
3	35	37	75	0.01	Sleeping	Sleeping	Sleeping
4	35	37	75	0.01	Sleeping	Sleeping	Sleeping
5	31	37	75	0.01	Sitting	Standing	Sitting
6	30	37	75	0.01	Sitting	Sitting	Sitting
7	29	37	75	0.01	Sitting	Standing	Sitting
8	28	37	75	0.01	Sitting	Sitting	Sitting
9	27	37	77	0.01	Sitting	Standing	Sitting
10	26	37	79	0.01	Standing	Sitting	Standing
11	25	37	81	0.02	Standing	Standing	Standing
12	24	37	83	0.04	Standing	Sitting	Standing
13	23	37	85	0.06	Standing	Standing	Standing
14	22	37	87	0.08	Standing	Sitting	Standing
15	21	37	89	0.10	Standing	Standing	Standing
16	20	37	91	0.12	Standing	Sitting	Standing
17	19	37	93	0.14	Standing	Standing	Standing
18	18	37	95	0.16	Walking	Walking	Walking
19	18	37	95	0.18	Walking	Walking	Walking
20	18	37	95	0.20	Walking	Walking	Walking
21	18	37	95	0.22	Walking	Walking	Walking
22	18	37	95	0.24	Walking	Walking	Walking
23	18	37	95	0.26	Walking	Walking	Walking
24	18	37.1	100	0.28	Walking	Walking	Walking
25	18	37.1	110	0.30	Walking	Walking	Walking
26	18	37.1	120	0.32	Walking	Walking	Walking
27	18	37.1	125	0.35	Walking	Walking	Walking
28	18	37.1	130	0.40	Walking	Walking	Walking
29	18	37.1	140	0.50	Running	Running	Running
30	18	37.1	150	0.60	Running	Running	Running
31	18	37.1	155	0.70	Running	Running	Running
32	18	37.1	157	0.75	Running	Running	Running
33	18	37.1	160	0.80	Running	Running	Running
34	18	37.1	155	0.75	Running	Running	Running
35	18	37.1	150	0.60	Running	Running	Running
36	18	37.1	145	0.50	Running	Running	Running
37	18	37.1	143	0.40	Walking	Walking	Walking
38	18	37.1	135	0.35	Walking	Walking	Walking
39	18	37.1	130	0.30	Walking	Walking	Walking
40	18	37	120	0.25	Walking	Walking	Walking
41	18	37	110	0.20	Walking	Walking	Walking
42	18	37	100	0.15	Walking	Walking	Walking
43	20	37	95	0.15	Standing	Standing	Standing
44	25	37	85	0.08	Sitting	Sitting	Sitting
45	27	37	80	0.08	Sitting	Standing	Sitting
46	28	37	80	0.08	Sitting	Sitting	Sitting
47	28	37	80	0.06	Sitting	Standing	Sitting
48	28	37	75	0.05	Sitting	Sitting	Sitting
49	28	37	75	0.05	Sitting	Standing	Sitting
50	28	37	75	0.05	Sitting	Sitting	Sitting

As both Table II and Figure 19 show, the state determining with the combined HMM and FIS is, in the Forward-Backward algorithm implementation, adequate and as expected given the parameters and rules as defined. With the employment of the Viterbi algorithm however – which can adjust the entire determined state sequence at each step – the final sequence (that displayed in both the table and figure) appears erratic in the transitional stages of *Sitting* and *Standing*. As the states are being determined, the output for these instances is actually always *Sitting* rather than alternating between the two; it is only as the next state occurs that the previous is amended to being *Standing* in every other instance. This is due to the maximum probability and likelihood as determined by the Viterbi method that the sequence of observations is best explained by these states transitioning back-and-forth between each other given their transition probabilities, which in this test case could be too similar to avoid the repetition between the two. It does however show that at this test stage if a user were to exhibit such behaviour for real, that the two sequence determining methods disagree, therefore indicating the possibility that the behaviour sequence is unusual given the parameters: something considered in the later error detection chapter of this work.

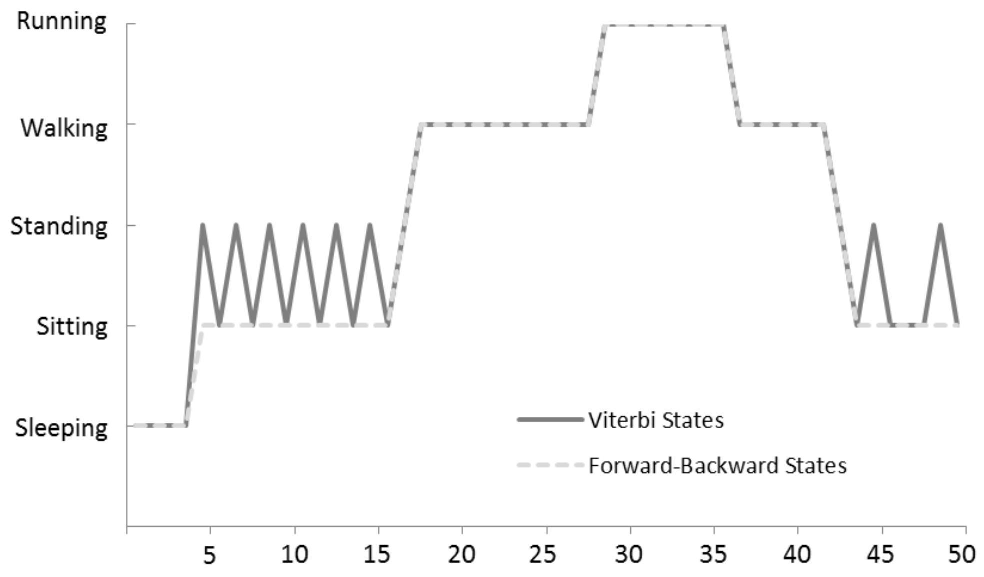


Figure 19 Results with FIS and HMM with above parameters and observations

With the FIS constructed as described and using the information programmed initially with human knowledge, the combined system is proven to be rather adequate in its behaviour detection. Utilising the fuzzy rule inference scheme however, we assume that for testing purposes the observation probabilities for each state based on the sensor reading combination have not yet been defined, and we aim to identify the membership functions and rules which best describe the activation of each state based on the above observation sequence by taking the "actual" observed state sequence as a supervision signal for training.

The input observations and readings from each sensor are processed as outlined above, determining the number of initial membership functions and the resolution required of each and the total size of the rule matrix. The test observation sequence generates an initial rule matrix of size:

$$|RuleMatrix| = number_{Ambient} \times number_{Contact} \times number_{Pulse} \times number_{Acceleration}$$

which is an 18×2×86×80 matrix, consisting of 247680 cells – though not all cells are active with states.

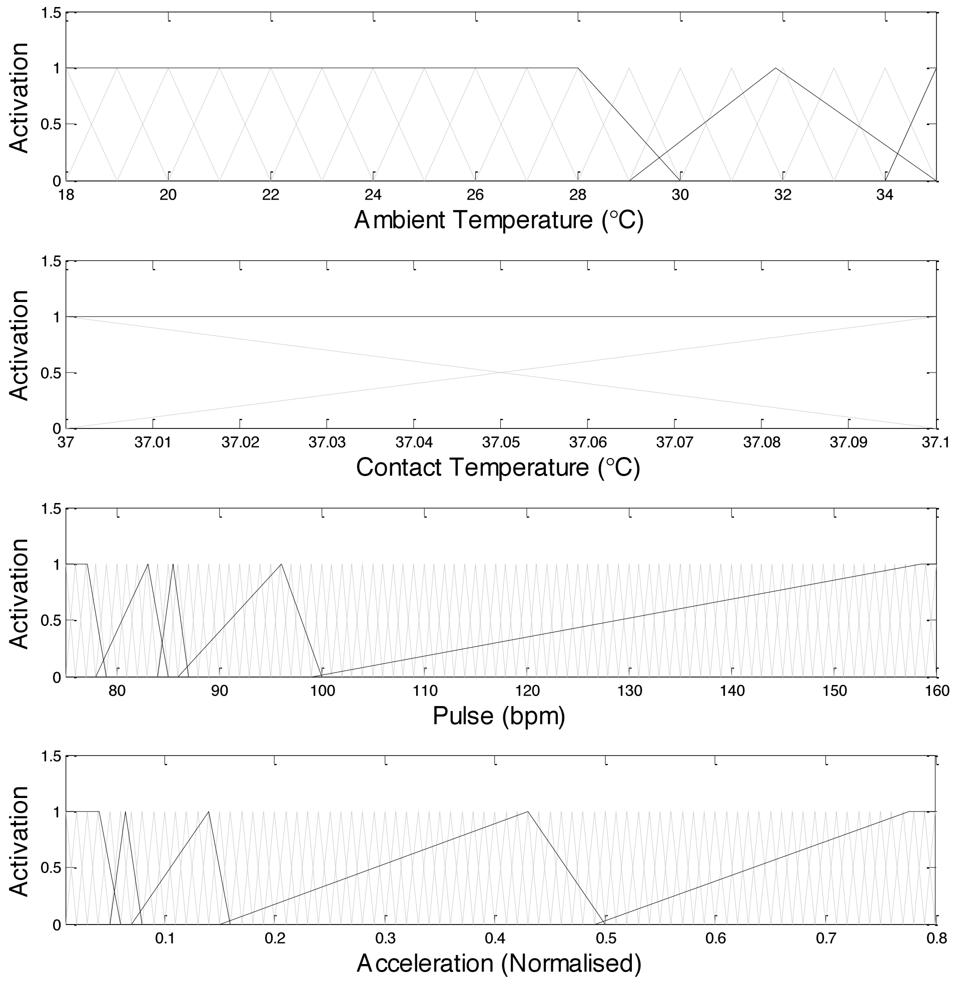


Figure 20 Resultant memberships after merging row-then-column

Submitting the rule matrix to the merge operations purely through a row-then-column approach, the resultant membership functions appear as in Figure 20, with rules inferred as per Table III. What is immediately apparent is that with the merging procedure the membership functions change considerably to accommodate only those observations witnessed as part of the input sequence, thereby removing irrelevant ones and creating others where necessary to better describe the relationship between observation and state.

Table III Rules inferred after merging row-then-column

New Rule	*IF*				*THEN*
	Ambient	*Contact*	*Pulse*	*Acceleration*	*State*
1	3	1	1	1	0
2	1	1	1	1	1
3	1	1	2	2	1
4	1	1	2	3	1
5	1	1	3	3	1
6	2	1	1	1	2
7	1	1	2	1	2
8	1	1	3	2	2
9	1	1	4	3	2
10	1	1	5	3	3
11	1	1	4	4	3
12	1	1	5	4	3
13	1	1	5	5	4

The number of membership functions for the ambient temperature input is the only one which matches the original FIS, however the central values and the overlap between adjacent memberships differs because of the lack of observations at the lower and upper bounds of the originals' range. The contact temperature is essentially removed from the FIS, inferring that it has no direct bearing on the determining of any of the states seen in the input sequence. The pulse reading input gains 2 extra membership ranges, with a significantly large range covering the 100-160bpm observations governing the Walking and Running states, and a smaller range spanning 84-87bpm which contributes to the determining of the *Sitting* and *Standing* states. Acceleration also gains 2 functions, with the rules indicating that extreme motion does indeed infer *Walking* and *Running*, but at the lower ranges it is not just acceleration governing the state decision.

Now, implementing the Unique Measure Merge described previously, the results differ as explained by Figure 21 and Table IV. Again, there are a few immediately apparent differences between the original FIS and this inferred one.

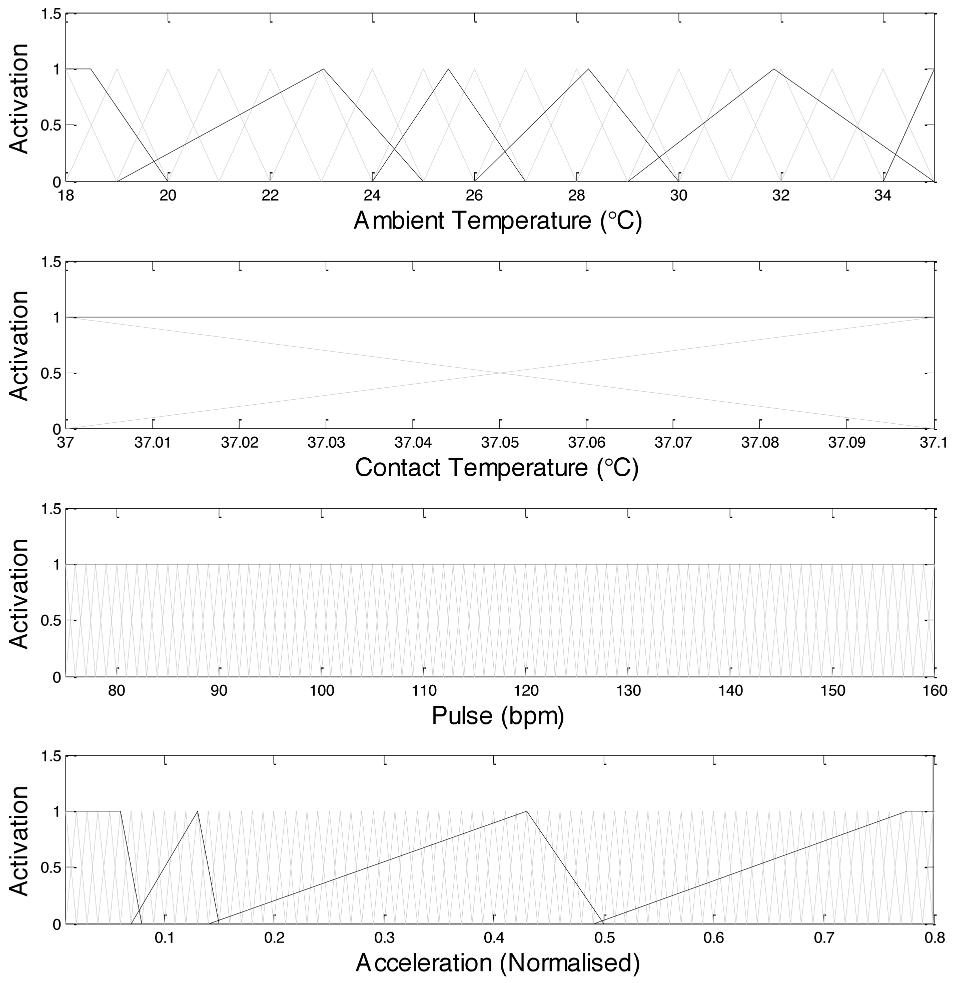


Figure 21 Resultant memberships after Unique Measure Merging

Table IV Rules inferred after Unique Measure Merging

New Rule	*IF*				*THEN*
	Ambient	*Contact*	*Pulse*	*Acceleration*	*State*
1	6	1	1	1	0
2	4	1	1	1	1
3	3	1	1	2	1
4	4	1	1	2	1
5	2	1	1	1	2
6	3	1	1	1	2
7	5	1	1	1	2
8	1	1	1	2	2
9	2	1	1	2	2
10	2	1	1	3	2
11	1	1	1	3	3
12	1	1	1	4	4

Firstly, it is now the case that 2 of the dimensions of the observation data have been deemed irrelevant to the state determining in the test case, with contact temperature and pulse reading now providing no differentiation support between any of the states – thus the state probability decisions rely on ambient temperature and acceleration alone. Ambient temperature gains

3 more membership ranges, which builds on the resolution of the previous rule inference to now help differentiate between all states - splitting the first membership range into 4 separate ranges; whereas acceleration's reduction of a range comes in its merge of the second function with the first to encompass values of 0-0.07 for acceleration in the test case. Overall, the change in order of row/column merging for each dimension has resulted in one less rule in the rule base, and in this case the effective removal of two of the dimensions of the data without a loss in the classification reliability of the FIS for observation probability determining.

Future observations however may not be explicitly accounted for with this approach, as not all possible outcomes have been considered in the inference of the rules purely with the supervision signal – this is using the instances provided by the training sequence to infer a relationship between observations and states and does not account for human knowledge that may be beneficial in future classifications. What the technique has provided is a means to adapt the system to a user after a considerable period of usage, where from a record of their observable signs and states can be inferred the values that are specific to them: a knowledge-based approach may assume that the "hot" temperature for a user's environment extends from 30° to 40°, but it may be more appropriate if their standard observable values for "hot" only ever reach 32° and anything seen subsequently over this value is deemed to be an indication of an error or abnormality, thus increasing the device's suitably to adapt to an individual.

4.5 Summary

A Fuzzy Inference System-incorporated approach has here been shown to greatly assist in fusion of multiple sensors for Hidden Markov Model-based behaviour recognition, when a single observation/state probability is needed in the Hidden Markov Model inferring. The detailed method computerises human knowledge into fuzzy rules and has been seen to be efficient in the observation probability parameter determining process, especially when data for the model is unavailable for traditional training methods and human knowledge more appropriately describes an observable process as in behaviour recognition. This combined approach provides scope for the future creation of models where there are a greater number of inputs determining a single output yet the model is still governed by human reasoning.

The development and subsequent inclusion of the rule inference process provides the basic Hidden Markov Model with the ability to operate autonomously, yet maintain an element of human reasoning in its modality of providing observation probabilities for combined sensor inputs. The rules inferred after a period which can be considered “training” are useful in the tailoring of the system to a specific user, removing irrelevant ranges and identifying values over which a state could be considered “erroneous”, and informing a healthcare professional of the operational bounds of each user for each predetermined state, thus becoming a useful diagnostic tool for any behaviour-related issues which may arise in time.

Chapter 5 Dimension Reduction and Classification

Fuzzy rule induction is ideal for adapting the Hidden Markov Model-based system to the user as more and more usage information becomes available. However, as the system expands to monitor more of the user with an increased variety of sensors, the data retrieved increases in dimensionality and the observations' relation to a state becomes increasingly nonlinear. The interpretation of such data using linguistic rules is now increasingly more difficult when operating directly in the high dimensional space, where the nonlinear manifold exhibits an increased complexity. Dimension reduction can unveil a lower-dimension manifold embedded within the high dimension data which explains the observation/state relationship more explicitly, allowing for better and simpler classification and visualisation.

The Curvilinear Distance Analysis for Linear Classification method is here developed to "unfold" the complex nonlinear manifold embedded in the multi-dimensional data into a lower dimension, where it becomes linearly separable. This lower dimension data affords a simple linear classifier – trained on the newly de-embedded manifold – the ability to always classify successive data from the same system.

5.1 Principle

The measurement of straight-line (Euclidean) distance within high dimension data is unreliable given its tendency to misrepresent true topology – in a high dimension space, data may appear clustered, yet when viewed from a single dimension the same data may actually be quite sparsely distributed or in fact consist of many component clusters. Similarly, identifying linear correlations (based on a least-squares approach) in the data as in Principle Component Analysis (PCA) [93 , 147] is not guaranteed to be accurate in nonlinear high dimension data given that such a distance relationship is unreliable. When reducing dimensions, it is therefore feasible to assume that reconstructing data purely with such a distance is inadequate and will result in similarly inseparable clusters as in the higher dimension space.

At present, nonlinear dimension reduction methods approach de-embedding problems in much the same manner as outlined here: taking the high dimension data’s clusters as separate entities within a singular manifold and linking them through the most logical prototypes of data so as not to stretch the resultant projection. This is exemplified by Meng, Leung and Xu [148] who succeed in high dimension manifold learning, by developing a method of finding the approximate edges of clusters in the high dimension and creating a “passage” between them which does not cut through the intrinsic manifold, instead smoothly connecting clusters such that 2D projection results in linear separation. Rosman *et al.* [149] similarly claim to improve on the Isomap technique, ignoring a geodesic distance as a linkage between points when it is inconsistent with Isomap’s assumption of convexity in the high dimension and resulting in projections which more appropriately describe data with “arbitrary boundaries and ‘holes’”. Though not explicitly reducing dimensionality for the purposes of classification, projection to a lower dimension with the method provides an ability to further process data more accurately as it still contains the key features embedded within it.

Within the original high dimension data, an embedded manifold of interest may appear classifiable by a Euclidean metric within its constituent clusters, however when each cluster is considered as a part of a whole high dimension manifold the Euclidean distance can fail to accurately represent similarity between adjacent values – being more appropriately described by a global distance measure which considers the entire data space. Curvilinear distance is one such measure which accounts for the topology of a high dimension manifold where Euclidean does not; Figure 22 provides a comparison between the standard Euclidean approach to identifying point-to-point similarity, and the principle upon which Curvilinear distance operates: linking points within the data space rather than “through” what can be considered a complex manifold.

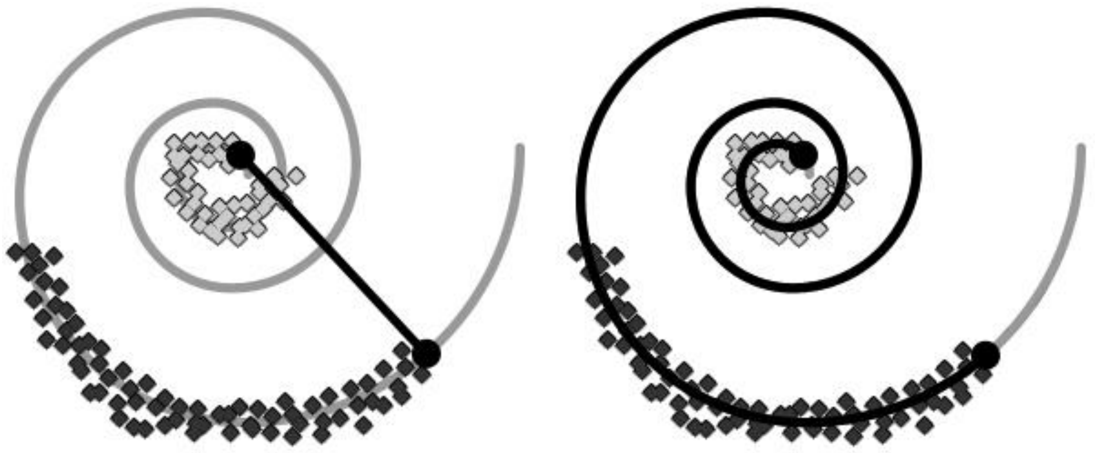


Figure 22 (a) Euclidean Distance measure "cuts" through a high dimension manifold, where a Curvilinear Distance measure (b) traverses the manifold to obtain a distance value

The Curvilinear Distance Analysis (CDA) technique of Lee *et al.* [90] is considered here, yet significantly extended to incorporate a method of separating clusters while reducing the intrinsic dimension of the data to produce Curvilinear Distance Analysis for Linear Classification (CDALC). CDA is capable of preserving the higher dimension space’s data topology while also sufficiently reproducing the local environment when said data is projected to the lower dimension.

The scheme proposed takes as its input a multi-dimensional training data set, in which a combination of dimensional values signifies classification to a cluster. Within Verity, this is essentially the multi-dimensional observation vector and the cluster identified as the classifier is the hidden state of the HMM that produced it. Let $X = [x_1, x_2, \ldots, x_M]^T$ be the data set of observations, where $x_i \in R^N$ is in an N dimensional space. Without loss of generality, we know that there are two classes, $X_A \in R^{M1 \times N}$ and $X_B \in R^{M2 \times N}$ in

$X \in R^{M \times N}$, where $M = M1 + M2$. Because X_A and X_B are not linearly separable, a nonlinear boundary has to be determined by multiple layered neural networks or nonlinear kernels if the classification of points belonging to these states is carried out directly in the high dimension space. This can be a tedious trial-and-error process, especially with networks that operate with multiple variable parameters.

Processing each of the class clusters in turn, those points which typify the topology of the cluster are identified as “prototypes”. These prototypes are then interlinked according to a predetermined neighbourhood, before each cluster is then connected to another via a single link and a graph created detailing curvilinear distances between all prototypes in the data set. The set $X = [X_A, X_B]^T \in R^{M \times N}$ is projected to a lower dimension $[X_a, X_b]^T \in R^{M \times n}, n < N$, where these distances are recreated as Euclidean, thus “flattening” the high dimension data and linearly separating the clusters X_a and X_b in the lower dimensional space. Based on distance to their closest prototype, successive points can be interpolated efficiently and projected from the high to the low dimension and once separated a simple classifier, e.g. a single layer perceptron, can be used to identify their parent cluster. This proposed approach is summarised in Figure 23.

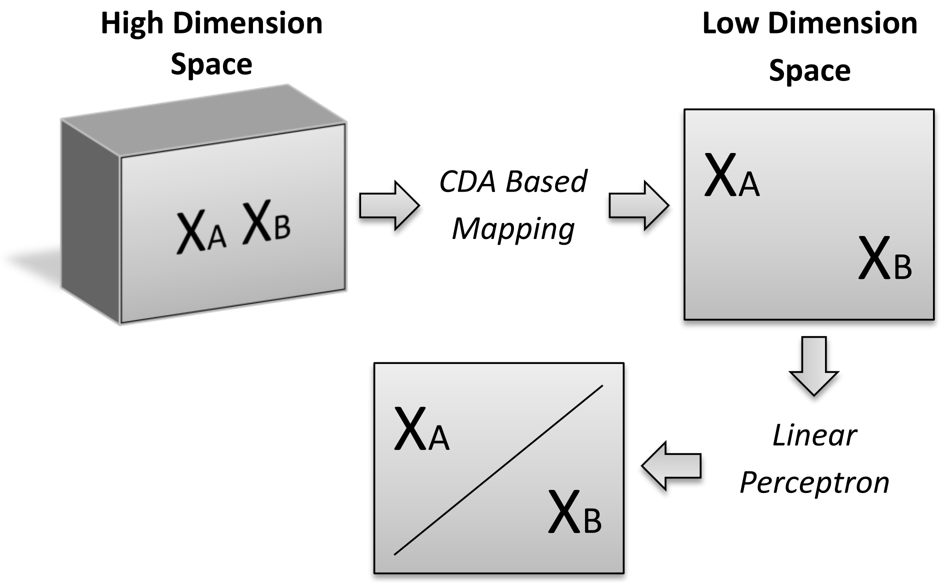


Figure 23 Dimension reduction with subsequent classification

The CDALC method is shown to be capable (through our testing) of successfully reducing up to 13 dimensions of data to 2 for successive classification, however with all data sets the most appropriate dimension in which to project varies depending on the influence of each original dimension. Here we assume an equal weighting on the contribution of each dimension of the Verity sensor data due to their constant variation and identify an appropriate lower dimension to be 2: affording us the ability to view the interaction between the states embedded in the 4D space and train our linear perceptrons on simple 2D data.

The CDA method of [90] was itself a further extension to the technique of Curvilinear Component Analysis (CCA) [91], and serves to unfold the data from high-dimension *n*-space to the low-dimension *p*-space. The distances between prototypes of the single manifold are kept, whilst the remaining data points from within that manifold are projected such that their *p*-space distance is comparable to the *n*-space distance about the closest prototype. The principle is used extensively to reduce single 3D manifolds to 2D, in reduction of dimensions of visual data [89] and in Mass Spectrometry [150].

Figure 24 shows a simple, general example of how in 2D, the data clusters (letters, in this case) are more easily identifiable than in the 3D representation. If the original 3D data were to be used in a supervised learning scheme for classification, the properties of each cluster would be unclear as some overlap between letters occurs. In such cases where this sort of data has been used for training, subsequent inputs are sometimes incorrectly classified when their properties place them within the overlap. If the dimensionally reduced data is used there is a higher likelihood of correct classification given that during training the separate cluster sets are able to be identified as such, with each having significantly different properties to another.

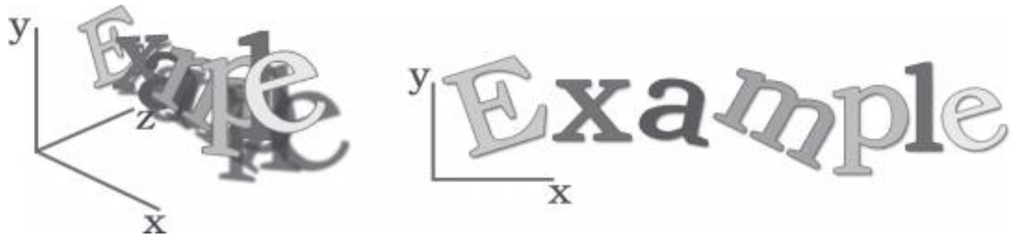


Figure 24 Reduction from 3D to 2D space results in better identification and separation

The documented CDA method is not equipped to tackle classification as required by the Verity system defined here. If data consisting of multiple clusters is treated as a single combined manifold, the properties which make a cluster unique can be lost in its projection to a lower dimension. Treating the clusters as separate homogenous manifolds and then linking (chain-like) together as one before projecting enables internal cluster structure to be retained in the same way that the overall topology is in the conventional CDA.

As in CDA, the data clusters are first quantised to provide the multidimensional prototypes – the vectors typifying a cluster's structure. The combined view of all prototypes in each cluster together approximates the layout of the high dimension manifold, with connections between each informing of the overall structure of the data set and the linkage distance between the prototyped clusters. A graph is created using these prototypes as the graph nodes, with the curvilinear distance calculated as the distance between prototypes using a standard routing algorithm (Dijkstra is the favoured one in this case [151]) and is therefore the total distance traversed along the path which connects them. The distance between each is such that it enables retention of global topology between dimensions once projected, thereby appearing as an "unfolding" of a manifold into the lower dimension.

Whilst in CDA the interpolation of successive unseen data points uses the common Euclidean distance to project locally (the shortcomings of which when used in a high dimension were discussed above and in the literature review of Chapter 2), a curvilinear distance more adequately represents a relation between data points in clusters as it enables visible separation over long distances and retains a better global topology. In this scheme the projection of supplementary interpolated points is therefore computed according to curvilinear distance also, given that CDA's employed Euclidean distance measure for local representation would be impossible to utilise effectively after the interconnection with curvilinear distance; this global distance is theoretically longer than Euclidean, therefore requiring the "stretching" of links as described in Chapter 2, with parasitic connections resulting which misrepresent the true topology of the manifold.

The order and basis of steps in this method follow those set forth by Lee *et al.* [89], however for the purposes of linear classification we modify the quantisation, prototyping and interlinking stages to ensure the high dimension manifold achieves maximum separation when projected in the similarly modified projection step, thus creating the CDALC method. This implementation is intended to require as little involvement with the user as possible, so there are few adjustable parameters. Currently within the method the most significant variable parameter is the tolerable loss of the vector quantisation; a higher value results in the quantised points generalising the clusters to a higher degree and thus computation is lighter, yet the resultant representation after interpolation can be seen to be less accurate.

5.2 Implementation of CDALC

Figure 25 explains the proposed operation of the Dimension Reduction operation: the processing of the data to obtain the prototypes via quantisation allows for interlinking to appropriately describe the cluster layout, before each state cluster is itself interlinked and projected to the lower dimension where the prototypes provide landmarks for the subsequently retrieved data to ally themselves to for the classification process with a standard linear classifier.

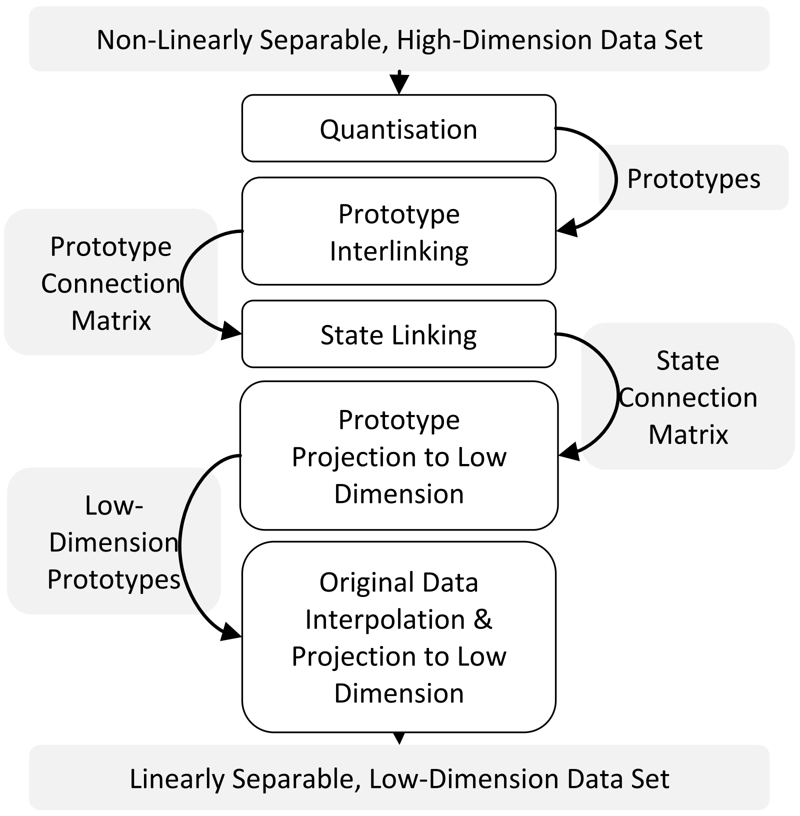

Non-Linearly Separable, High-Dimension Data Set
Quantisation
Prototypes
Prototype Interlinking
Prototype Connection Matrix
State Linking
State Connection Matrix
Prototype Projection to Low Dimension
Low-Dimension Prototypes
Original Data Interpolation & Projection to Low Dimension
Linearly Separable, Low-Dimension Data Set

Figure 25 The 5 step process to reduce the dimensionality of the data set

5.2.1 Quantisation

From experimentation it is found that raw, high dimension data is not always conducive to computationally fast operations. A distance relation in the *n*-space can be in the order of thousands, which when summing for the error over all data points can place great strain on a typical system. Normalising all input data to values between 0 and 1 first solves this problem, allowing for fair consideration of inter-dimensional distances (non-normalised data using a Euclidean distance measure can put unfair weighting on dimensions with larger scales) and ensures that no distance calculations between points should produce results of too high an order to be computationally tractable (i.e. maximum straight line distance in a unit space is $\sqrt{2}$). Given that the result in *p*-space is used solely for clustering and classification purposes and does not require attributing to a measurement or unit, the normalised distance representation will readily suffice.

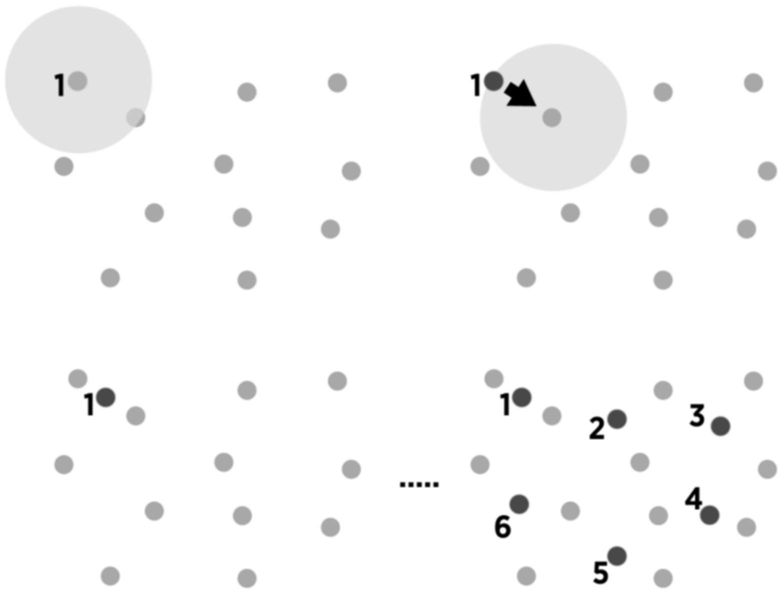


Figure 26 Prototype identification and tuning based on distances to local data points

A prototype is a data point in the *n*-space which will serve as a marker in the *p*-space around which data points can be projected. Using vector quantisation, the best prototypes representing these data sets can be determined. We approach each cluster individually and create prototypes within them in order to provide a decent generalisation of that cluster’s topology. Any vector quantisation method may be employed, where here a dynamic vector quantisation is utilised, which uses a competitive update scheme to bring the prototypes to a more reliable representation of the data.

Taking the first data point in the cluster, it is assessed for distance to each other point, with the maximum returned distance informing of the radius outside which a point is considered a prototype – by combining it with a $tolerable\_loss$ value. The $tolerable\_loss$ essentially determines how much of a bearing the maximum distance has on the neighbourhood, with a lesser $tolerable\_loss$ value meaning more determinable prototypes as fewer data points fall into the radius (as shown in Figure 26). Once the value for the radius has been set, each point in turn is assessed for its closeness to a prototype using the neighbourhood value: if the point falls within the radius of a prototype, that prototype moves to be closer to that point by a value set by $alpha$, or if it falls outside the radius it then becomes a new prototype itself to be considered by subsequent neighbourhood assessments.

In order to ensure the fair moving of the prototype between two or more constituent points, rather than working on the mean distance between all which would require constant re-evaluation of each vector, $alpha$ is a distance which decreases with every iteration from a maximum to a minimum value set by the programmer and acts on the difference between prototype P and point j:

$$alpha = d_{Pj} \times alpha_{\max} \times \left(\frac{alpha_{\min}}{alpha_{\max}} \right)^{\frac{iteration}{iteration_{\max}}} \tag{56}$$

The maximum number of iterations used in the quantisation, $iteration_{\max}$, determines the speed at which the prototype moves between the two vectors until such time as the maximum has been reached – when the distance is then equal to the minimum moveable distance possible, $alpha_{\min}$. When $iteration_{\max}$ has been achieved, the prototypes will describe the data as per Figure 26, having reduced the number of points to be considered by selecting those which best describe the data. The pseudo code below serves to further explain the quantisation process.

```
for each cluster
     max_distance is 0
     for all data points in cluster
      if distance between 2 points is greater than max_distance
      distance becomes max_distance
      end
     end
     radius = max_distance × tolerable_loss
     prototype = []
     prototype_num = 0
     while iteration is acceptable or prototype_num continues to
     increase
        for all data points in cluster
        for all prototypes
                  if data point is not within radius of prototype
           data point becomes a prototype
          prototype_num = prototype_num + 1
          else
                       move closest prototype within radius by
                       alpha_, an amount which decreases with
                       every iteration
          end
end
        end
     end
end
```

The convergence to an optimum number of prototypes will typically occur early on in the process provided the $tolerable\_loss$ and $alpha_{\max}$ values are suitably defined for the data set (typically both $[0,1]$) – it is at this iteration we can stop the vector quantisation or choose to continue on for the set number of iterations. This step is performed for each cluster, before the next step links each cluster internally for the proceeding interconnection throughout the manifold.

5.2.2 Prototype Interlinking

This step requires another initial neighbourhood radius to be set, which in this instance is the mean of the 3 shortest distances between all prototypes (to enable at least one point to be encompassed by the radius for the first iteration):

$$k_{init} = \frac{1}{3}\sum_{i=1}^{3} a_i \qquad (57)$$

Where a is an ordered list (from low to high) of distances between all prototypes in the cluster. The step-increase of the neighbourhood per iteration is then found with the initial neighbourhood and the maximum distance between prototypes:

$$k_{step} = \frac{k_{init}}{\max[a]} \tag{58}$$

A square linkage matrix (59) is also created which contains information about which prototypes within a cluster are connected. It consists of $[a]^2$ entries and is initially populated entirely by "Infinity" (∞) values to symbolise all are initially non-traversable links.

$$\begin{bmatrix} \infty & \cdots & \infty \\ \vdots & \ddots & \vdots \\ \infty & \cdots & \infty \end{bmatrix} \tag{59}$$

When evaluating each individual prototype i, if another, j, falls within the neighbourhood radius then the linkage matrix updates to accommodate the distance between the two at that index point ($link_{ij}$). Once the neighbourhood is evaluated, Dijkstra's algorithm is employed to ascertain whether or not prototypes can now reach all other prototypes through the created linkages. In the initial instance with the first prototype this will obviously be impossible; we decide if all linkages are traversable by returning the maximum distance in the Dijkstra matrix: if the maximum distance remains "∞" there are still some unreachable prototypes. If some are still unreachable, the neighbourhood radius increases by a step-size and the evaluation process continues until the maximum distance is reduced. Once the linkages are all traversable, it can be said that the cluster is fully connected. The process repeats for all clusters until they are all internally linked.

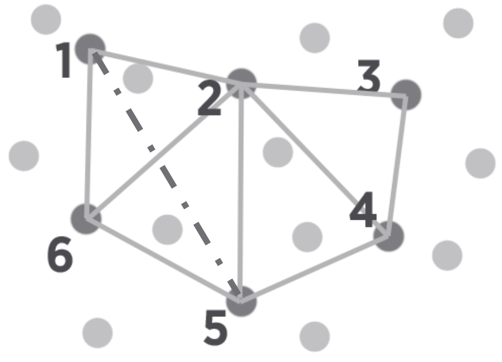
The linkage matrix at each stage updates with the first calculated distance between prototypes: if the neighbourhood radius happens to include a prototype via the calculated Euclidean distance before it is reachable via curvilinear distance through the Dijkstra routing algorithm, the Euclidean distance becomes the shortest from one prototype to another. However, if the curvilinear encompasses a prototype before the neighbourhood expands, then that distance is the one considered the shortest. This avoids a web-like connection matrix which can cause subsequent projection errors.

```
neighbourhood is average of 2 smallest Euclidean distances between
prototypes
step is neighbourhood ÷ max Euclidean distance between prototypes

while (max distance between all prototypes determined through
Dijkstra is infinite)
      for all prototypes in state cluster
            for all prototypes in state cluster
                  if (distance between both prototypes is less
                  than neighbourhood)
                        if (prototypes not already connected in
                        Dijkstra graph)
 both prototypes linked
                        end
end
end
end
      neighbourhood is neighbourhood + step
end
```

CDA originally used a *k*-Nearest Neighbours approach to internal linking which sometimes would result in a parasitic connection between parts of a manifold. This parasitic connection could result in the unfolded manifold having two distant prototypes connected by a single link spanning the entire projection, thus being an unrealistic representation of the data set. The automated method here is more suited to the CDA process for classification than the documented original as it removes the possibility of such connections. Through use of the Dijkstra algorithm, constantly assessing the cluster for already apparent connections via a Euclidean distance, the best representation of curvilinear distance is found without compromising a cluster's topology by cutting through paths to other prototypes. A trade-off in this enhanced representation of a cluster however is the increased time consumption experienced from the repeat employment of the routing algorithm not shared by the standard *k*-NN approach CDA, as it requires all prototypes to be considered during every iteration for the determining of the presence of a path between them rather than linking solely according to prototype neighbours falling within a fixed radius.

Figure 27 visualises the connection between prototypes within a state cluster. Whilst in an iteration of the linking process prototypes 1 and 5 would fall within the same radius, as can be seen on the adjacent matrix a connection already existed between the two – in this case by travelling via prototype 2 or prototype 6 would allow for arrival at prototype 5. If we didn't use this curvilinear distance method, the Euclidean connections between 1 and 5 (shown as the dotted line) and 2 and 6 would cause projection errors, as they cross paths in order to reach their destination prototypes. An attempt at replication of this distance would result in stretching and tearing of the cluster manifold in order to obtain a minimised projection error.



	1	2	3	4	5	6
1	✓	✓				✓
2	✓	✓	✓	✓	✓	✓
3		✓	✓	✓		
4		✓	✓	✓	✓	
5		✓		✓	✓	✓
6	✓	✓			✓	✓

Figure 27 Internal connections within a cluster and the prototype linkages from one to another. The dotted line shows the Euclidean variation of traversal from point 1 to 5 which ignores the already computed Curvilinear distance via points 6 or 2

5.2.3 State Linking

In order to represent the clusters in a lower (unfolded) dimension such that the topology of the higher dimension is retained – yet maximum separation achieved for linear classification – the clusters are linked through their furthest distance from each other. That is to say that each cluster, once internally linked, has each of its prototypes evaluated for its Euclidean distance to all other prototypes in other clusters. The maximum distance from one cluster to another (i.e. furthest distance between two prototypes of different clusters) then becomes the minimum linking distance and therefore the shortest distance present between clusters in the lower dimension. Figure 28 demonstrates the linking, with clusters 1 through 5 linked independently to their closest neighbour forming a chain link from the first to the last.

The output of this step is a matrix of distances between prototypes of the data set. The matrix effectively forms a "map" of how far apart each prototype should be from each other now that they are all linked in such a way as to be viewed as a "roll" or a "folded" plane. This means that once projected to the lower dimension, there should be no concerns regarding stretching of linkages or tearing of the plane, as it is just an unfolding of prototypes from a higher dimension.

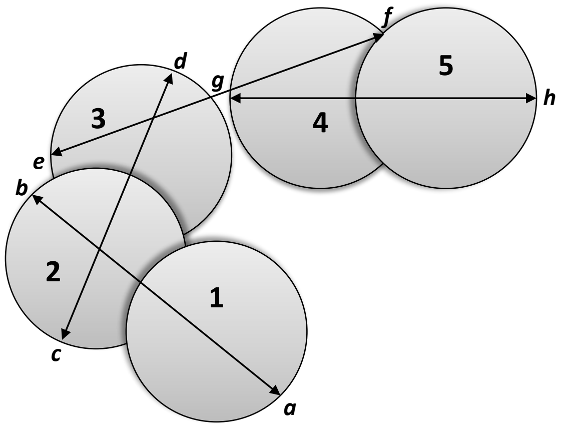


Figure 28 Linking of the clusters by maximum distance: 1 to 2 via *ab*, 2 to 3 via *cd*, 3 to 4 via *ef* and 4 to 5 via *gh*

5.2.4 Prototype Projection to Low Dimension

The determining of the curvilinear distances between all prototypes in all clusters occurs as in the original CDA implementation, using Dijkstra's algorithm to calculate and produce a matrix which contains all pairwise curvilinear distances for the data set.

The projection involves reducing the error between the computed curvilinear distances of the *n*-space and the new equivalent Euclidean distances in the lower dimension *p*-space. The result is an "unfolded" or "flattened" representation of the original *n*-space data which retains the distances between prototypes of a cluster yet separation between clusters has been maximised.

The error function sought to be minimised is:

$$E_{CDA} = \sum_{i,j=1}^{n} \left(\delta_{i,j}^{n} - d_{i,j}^{p}\right)^{2} F\left(d_{i,j}^{p}, \lambda\right) \tag{60}$$

Where above, $\delta_{i,j}^{n}$ is the calculated curvilinear distance in *n*-space and $d_{i,j}^{p}$ is the Euclidean distance in *p*-space between prototypes i and j. The function F weighs the contribution of the prototype pair so that local reproduction is favoured over global – thus the larger the distance in *p*-space over *n*-space the less of a bearing it has on the error and projection and varies between 0 and 1 when its argument increases and decreases respectively over time. It is intended that this factor enable the CDA method to reproduce more effectively the smaller distances over larger ones within the manifold.

$$F\left(d_{i,j}^{p}, \lambda\right) = \begin{cases} 0, \lambda - d_{i,j}^{p} < 0 \\ 1, \lambda - d_{i,j}^{p} \geq 0 \end{cases} \tag{61}$$

The neighbourhood distance value of λ must be large enough to allow fast convergence [89] and suitable representation of the high dimension data when projected, yet small enough to avoid preserving long Euclidean Distances when the projected manifold is nonlinear. Higher λ values consider more distances in the low dimension *p*-space, thus increasing the accuracy of projection in the global area over the local area. This means it must be initially higher than the maximum value in the $\delta_{i,j}^{n}$ matrix in order to unfold the *n*-space manifold correctly on a global scale before addressing the local projections, as the value of λ decreases every iteration and therefore optimises smaller distance relationships:

$$\lambda = \lambda_{\max} \times \left(\frac{\lambda_{\min}}{\lambda_{\max}}^{iteration \times \lambda_{\min}} \right) \tag{62}$$

The initial projection of the prototypes is random in the lower dimension *p*-space. The distance between each prototype is calculated as in the previous step, except now in this lower dimension (which is much quicker and less computationally complex given the difference in dimension vectors). The result is another distance matrix which is comparable to that previously computed, with the intention now to move these projected prototypes about each other in order to match the high dimension distances with as little error as possible. The above equation (60) is minimised to the point of acceptable error, at which point the moving of the prototypes ceases.

In order to minimise the projection error of the prototypes from the high dimension to the low dimension via gradient descent of (60), all but one prototype are moved in a single iteration of the relocation step; instead of the usual method of updating the projections where a single point moves according to the position of others, this procedure holds a point z_i stationary and moves others $z_{j \neq i}$ radially around it. Computationally, this adaptation process for the location of prototypes is much lighter than a standard gradient descent method, as it is not necessary for the computation of all $N(N-1)/2$ distances in the *n*- and *p*-spaces, just the distance between prototype z_i and all others [91]:

$$z_j \leftarrow z_j + \alpha F\left(d_{i,j}^p, \lambda\right)\left(\delta_{i,j}^n - d_{i,j}^p\right)\frac{z_j - z_i}{d_{i,j}^p} \tag{63}$$

The value of α, the learning rate, is usually time-decreasing to ensure a convergence:

$$\alpha = \frac{\alpha_0}{1+t} \tag{64}$$

After the acceptable adjustment of the prototypes about z_i according to equations (63) and (64) for one iteration, the distance $d_{i,j}^p$ is recalculated for the movement of the next prototype in the set. Once all prototypes have been located and their placement converges to a minimum (local or global), the projection error is then calculated with equation (60). Rather than wait until convergence for large sets, however, the option to cease relocation of prototypes can be taken once the projection error has reached a suitably low value as defined at the outset by its recalculation periodically throughout the process.

5.2.5 Original Data Interpolation and Projection to Low Dimension

Once the initial data prototypes have been acceptably located in the lower dimension, the original sensor observation data which these prototypes describe must be similarly placed also. Given that the prototypes represent the data's clusters, their projections in the *p*-space can be taken as landmarks around which to project these other points from within the data set.

Given that for this developed algorithm the linkage between clusters in the high dimension is a single connection, the resultant representation in *p*-space is required to be somewhat stretched in order to maintain the curvilinear distances calculated with as little error as possible. This forces the original point interpolation and projection scheme developed for CDA to be further modified to accommodate the change. As a result, the "local", direct mapping of point-to-prototype Euclidean distance is replaced with a "global" mapping using the curvilinear distances (which may be significantly larger than Euclidean given the traversal of prototypes) - as was used in the projection of the original prototypes. This means that instead of locating the *x*-closest prototypes to a data point and projecting in *p*-space according to their Euclidean distance to them (as in the original CDA), we first identify the 3 closest prototypes according to their Euclidean distance. Had just 1 prototype been taken as a reference location, the projection of the point may be anywhere 360º about the prototype; 2 prototypes would result in 2 possible positions whereas 3 allows for adequate triangulation of the interpolated point.

We typically determine the first closest prototype's distance to the other two not through the straight-line distance but the curvilinear, as it may be true that all closest prototypes belong to different clusters and therefore to represent locally (given the stretching of the links) is not possible. Using these curvilinear distances, as in the linking of the original prototypes, the data is embedded on the same multi-dimensional manifold and "unfolded" into the lower dimension with the prototypes, rather than having its projection compromised through usage of the Euclidean distance for a cross-state distance measurement which would stretch its placement as in a standard nearest neighbour approach.

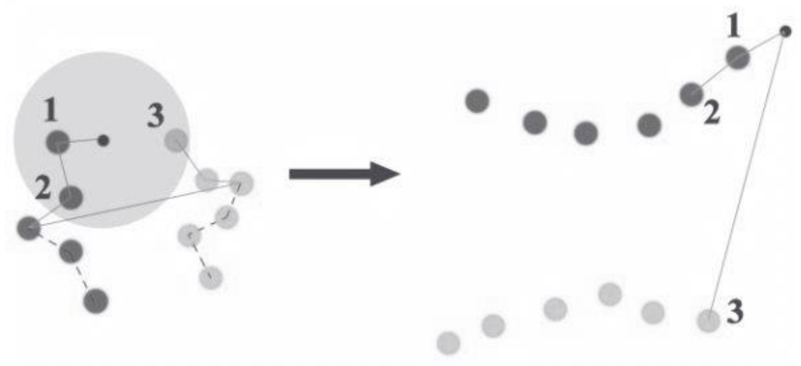


Figure 29 Selection of 3 closest prototypes to data point (with light grey neighbourhood) in higher dimension and the subsequent projections in the lower dimension

Figure 29 graphically explains the process of point interpolation and location identification. With the high dimension clusters' prototypes being internally linked, the connection between clusters is taken as the "bridge" which enables multiple cluster prototype positions to inform of the point's location. In the case of this figure, the point is closest to prototypes 1, 2 and 3 as they fall within its pre-defined radius. As can be seen in the resultant projection, the point is placed within the pattern of the darker cluster, yet its exact location determined by distance from the lighter cluster's prototype (with the label "3").

The optimization of the error function for the point projection works the same as with CDA, whereby a single point is moved in relation to all others in order to minimize the error between the high dimension curvilinear and the low dimension Euclidean distances to the closest prototypes. Instead of a global assessment for the error of the projection, we are only concerned with the local projection about the prototypes which describe the manifold and therefore N here represents the total number of neighbours used in the triangulation of location of the projected point (which in this case, would be 3 as explained above). For the interpolation the error function appears as:

$$E_{CDApoint} = \sum_{i,j=1}^{N} \left(\delta point_{i,j}^{n} - dpoint_{i,j}^{p} \right)^{2} \tag{65}$$

Given that now, the points are being moved one-at-a-time with respect to the prototypes, the relocation algorithm is updated to reflect this:

$$point \leftarrow point + \alpha\left(\delta point^{n} - dpoint^{p}\right)\frac{point - closestPrototype}{dpoint^{p}} \tag{66}$$

Again the values of $dpoint^{p}$ are recalculated after a relocation and this step repeats until the error (65) is suitably small for each projected point.

Once the original data set is projected, clusters are visible in the *p*-space as being linearly separable: so long as the maximum distance between clusters is large enough to overcome possible projection errors which may project prototypes in too close a proximity to others from different clusters.

It can be proven that a data set can always be made linearly separable with the proposed method if the distances from the high dimension space are retained; thus a single layer perceptron is sufficient for subsequent classification. Figure 30 describes how the FIS is replaced with the Dimension Reduction and Classification scheme (the grey box is effectively removed once its values are used in the reduction process) before its output is sent to the cascading perceptron network described below which differentiates between each state now that they are linearly separable.

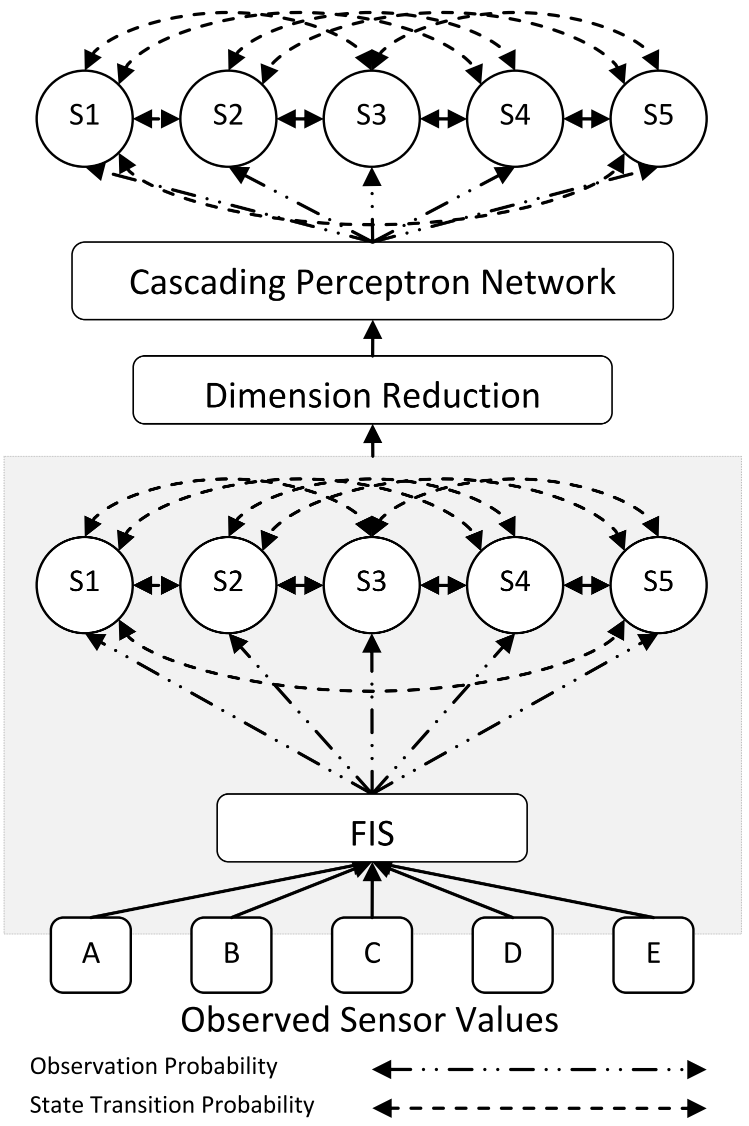


Figure 30 The new dimension reduction scheme which interacts within the HMM using the data from the original FIS

5.2.6 Linear Separability

A standard perceptron is a linear classifier, capable of predicting an observation vector's state as belonging to a binary class (Figure 31).

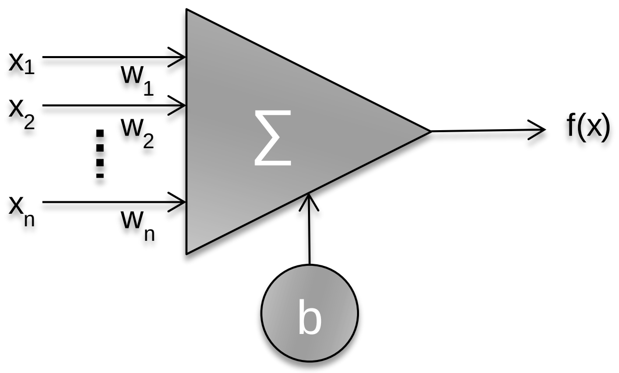


Figure 31 Basic perceptron construction, with input vectors x multiplied by weights w before the addition of the bias, b. The output is a binary value informing of the classification

The input observation is mapped to $f(x)$, a binary output:

$$f(x)=\begin{cases}1 & \text{if } w.x+b>0\\ 0 & \text{otherwise}\end{cases} \tag{67}$$

Within the perceptron, there are the vector weights, w, which act on the vector x and a single constant bias value b which is applied to the dot product with no regard for the input value.

For proof of the algorithm's suitability, we assume the state clusters X_A and X_B are mapped to a lower space to be X_a and X_b. It is theorised that X_a and X_b are linearly separable if the linking distance $d > diam(\mathrm{X}_a)+diam(\mathrm{X}_b)$, where $diam(\ )$ is the diameter of a cluster, i.e. the maximum in-cluster distance in the completed distance matrix (59).

The furthest points for X_A and X_B are labelled as x_A and x_B in the high dimension. They are projected to the low dimension space as x_a and x_b. A perceptron can therefore be constructed with weight and bias:

$$W=\frac{(x_b-x_a)}{\|x_b-x_a\|} \tag{68}$$

$$b=-x_a^{\ T}\frac{x_b-x_a}{\|x_b-x_a\|}-diam(X_a) \tag{69}$$

The line function for classification can be written as:

$$net=W^T x_i+b=\frac{x_b-x_a}{\|x_b-x_a\|}(x_i-x_a)-diam(X_a) \tag{70}$$

With the perceptron output $f(x)=\mathrm{sgn}(net)$. For any sample x_{ai} taken from cluster *A,* the net output can be obtained from (70):

$$net(x_{ai})=\frac{(x_b-x_a)^T}{\|x_b-x_a\|}(x_i-x_a)-diam(X_a)\le\|x_{ai}-x_a\|-diam(X_A)<0 \tag{71}$$

Therefore the perceptron output $f(x_{ai})=0$.

For any sample x_{bi} taken from cluster *B*, the net output can again be obtained from (70):

$$\begin{aligned} net(x_{bi}) &= \frac{(x_b-x_a)^T}{\|x_b-x_a\|}(x_{bi}-x_a)-diam(X_a)\\ &= \frac{(x_b-x_a)^T}{\|x_b-x_a\|}(x_{bi}-x_b+x_b-x_a)-diam(X_a)\\ &= \frac{(x_b-x_a)^T}{\|x_b-x_a\|}(x_{bi}-x_b)+\|x_b-x_a\|-diam(X_a)\end{aligned} \tag{72}$$

Given that:

$$d = \|x_b - x_a\| > diam(X_a) + diam(X_b) \tag{73}$$

Then it stands to reason that:

$$net(x_{bi}) > \frac{(x_b - x_a)^T}{\|x_b - x_a\|}(x_{bi} - x_b) + diam(X_b) \geq diam(X_b) - \frac{\|x_b - x_a\|}{\|x_b - x_a\|}\|x_{bi} - x_b\| \tag{74}$$
$$= diam(X_b) - \|x_{bi} - x_b\| \geq 0$$

Therefore the perceptron output $f(x_{bi}) = 1$. The projection in the low dimension space is linearly separable. Using the perceptron as a simple linear classifier, it has been proven that the two originally high dimension-embedded, nonlinear classes become linearly separable once projected to the lower dimension – provided that the linking distance between them is longer than the sum of their within-cluster diameters, which is always the case if the two furthest prototypes in each cluster are used as connecting nodes for the global curvilinear distance calculations. As a result, we can employ the perceptron as a classifier for the once high dimension data that was intrinsically nonlinear.

For the scheme used here in the HMM, a cascading network of perceptrons is employed to differentiate one state from all others as a means of simply classifying the observation data in the low dimension. Figure 32 shows the principle method of distinguishing 5 states from each other utilising 4 perceptrons, as is the case for the Verity application.

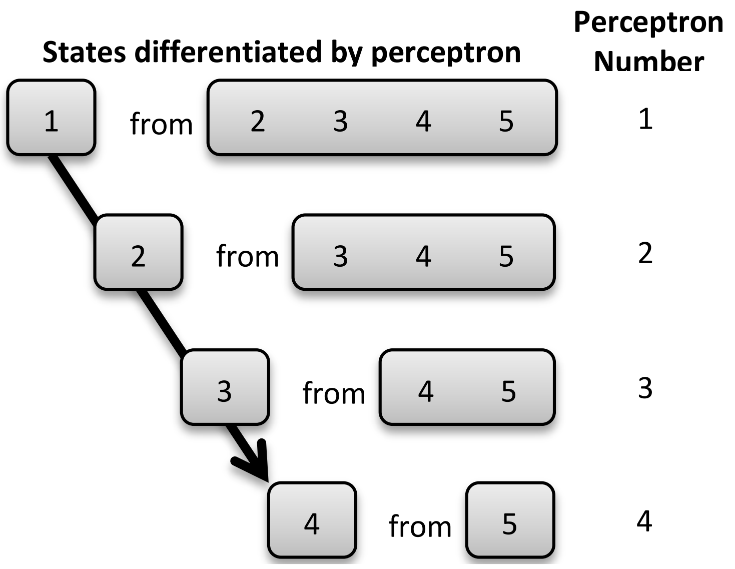


Figure 32 Cascading Perceptron Network differentiating between 5 different states

5.3 Common Data Set Examples

Many data sets exist for the testing of classification methods and dimension reduction techniques. Commonly the most difficult problem is in representing and separating nonlinearly separable data. The publically available data sets used here to illustrate suitability and effectiveness of the algorithm are *Fisher's Iris Data*, *Bupa Liver Data*, *Breast Cancer Data* and *Wine Data* – all of which are available from the UCI Machine Learning Repository [152] for such purposes. A 3 dimensional set of artificial test data was also created to aid in the visualisation of the reduction method, given that the other sets are impossible to project (for visualisation purposes) in their original high dimensional form.

All 5 data sequences are split into two separate sets, with one set used in the prototyping and projection and the other for successive testing. The dimensionality of each data set varies significantly, with the maximum being 13 for the Wine Data and the minimum being the 3D artificial data.

The results of each reduction can be seen in Figure 33 to Figure 38, where the resultant projection of successive data is promising yet obviously not without error (discussed in the next section). The prototypes of the test data are projected as unbounded circles, with the interpolated points from both the initial test sequence and the successive data being distinguishable by their darker, bounded edges.

Using the curvilinear dimension reduction method, placement is optimised to the point where the overall error between recreated distance in the lower and actual distance in the higher dimension is reduced to less than 0.1. In practice this results in reasonable representations of the original data, whilst not being too computationally time consuming. If the application requires it, the maximum number of iterations of error reduction can be set – however to achieve optimal representation this would require other parameters within the CDA method to be adjusted to accommodate the change.

It can be clearly seen that the each of the clusters within the original data set is reproduced separately and without overlap in the lower dimension, and the interpolated points also fit within the respective boundaries laid out by their prototypes – thus providing the linear separation required for the application of the neural network or other such similar procedure.

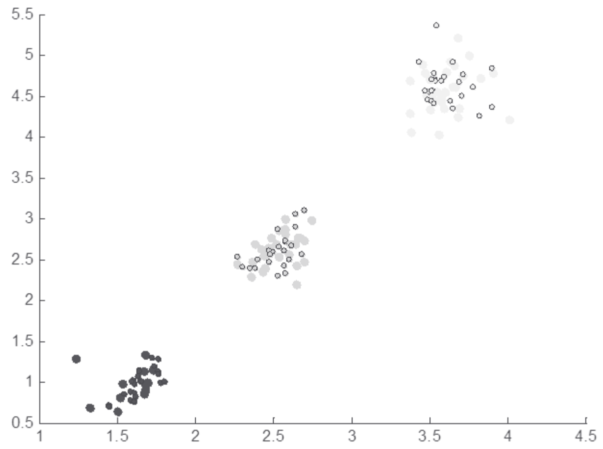


Figure 33 Dimension Reduction Algorithm applied to Fisher's Iris Data

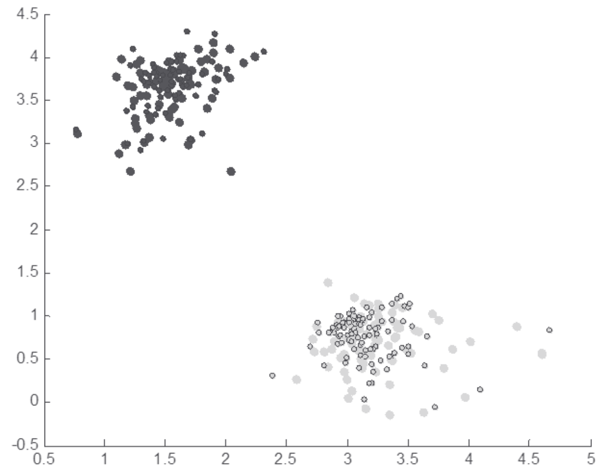


Figure 34 Dimension Reduction Algorithm applied to Bupa Liver Data

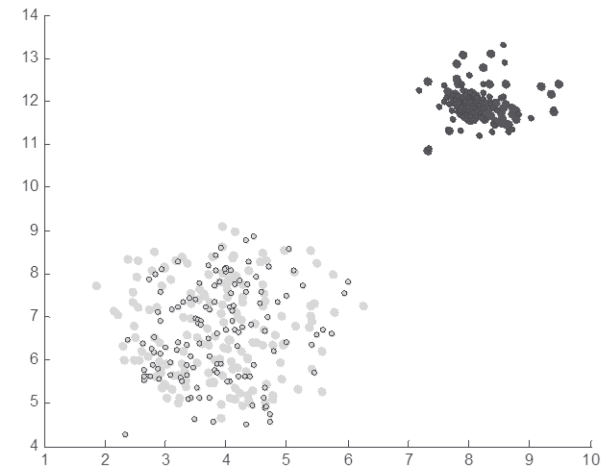


Figure 35 Dimension Reduction Algorithm applied to Breast Cancer Data

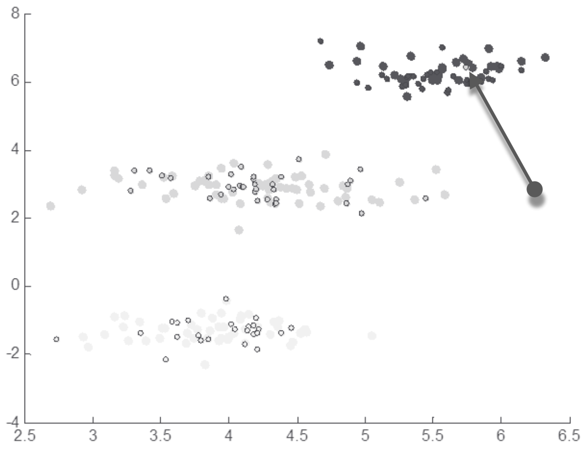


Figure 36 Dimension Reduction Algorithm applied to Wine Data. [Arrow explained in “Testing and Results”]

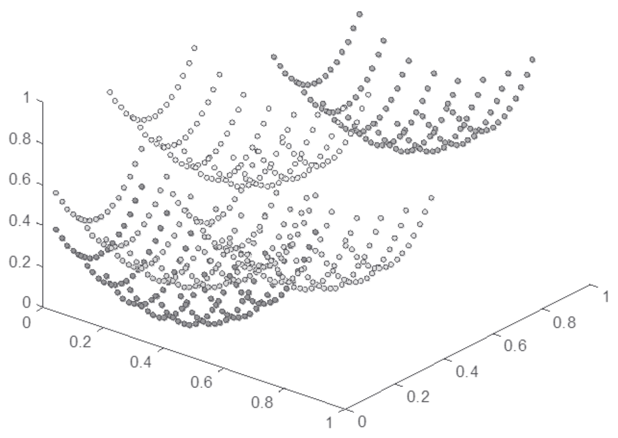


Figure 37 Artificial 3D data created for testing purposes. Note the nonlinear properties of each "bowl" overlapping another

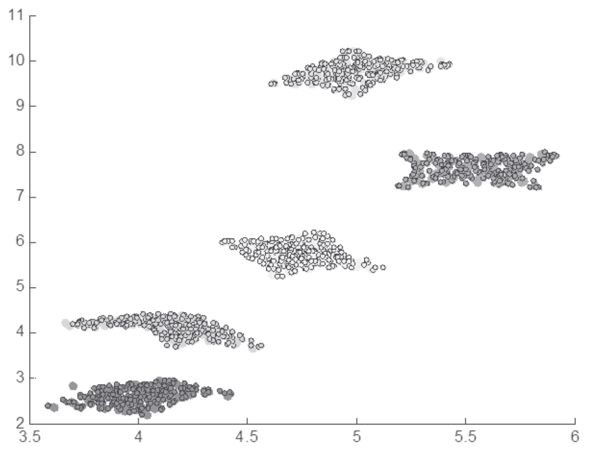


Figure 38 Dimension Reduction Algorithm applied to Artificial 3D data. Note now the separation between each "bowl" and the correct classification of each data point within

5.3.1 Testing and Results

Whilst it is evident that the clusters are adequately represented as linearly separate in the lower dimension, it is also possible to see that in some data sets there will be misclassifications of some of the successive, unseen data points. Given that the neighbourhood radius is used to determine the membership to a cluster, if a prototype falls too close to a point which does not belong to that cluster, the point immediately associates itself with that cluster. This is evident in the Wine Data set where one point which would normally belong to the middle class has in fact been identified as being a member of the top class (indicated by the arrow present in Figure 36). A different means of "closest" prototype selection (more prototypes informing of the closest class prototype, or perhaps average distance) may in fact place the wrongly-located point in the correct class, however it may be at further cost to other point placements within the projection.

What should be noted is that within Fisher's Iris Data (Figure 33), the two nonlinearly separable clusters of Iris Virginica and Versicolor have been separated and the interpolated points have been successfully tagged to the correct cluster with no such error. With many supervised learning techniques this would also be possible, however through prototyping the initial clusters we have achieved a generalised view of each set, to which the neighbourhood function of successive points adequately allows for correct interpolation.

Using Matlab once more for its rapid visualisation ability, the effectiveness of the dimension reduction/clustering can be tested using a comparable neural network approach. A feed-forward back-propagation network is capable of solving almost any problem provided the network parameters are correctly chosen. In this test the Iris data in its raw form is submitted to the network and training commenced. The network consists of 2 layers and 20 neurons in the input layer and is trained for 1000 epochs or until the generalisation stops improving. Then the pre-processed, dimension reduced data is submitted to a similarly constructed 2 layer network.

With the non-reduced data, the classifications are returned as values which must be taken to one significant figure to provide a class; the reduced data provides correct classes of 1, 2 or 3 straight from the network. In the case of the non-reduced data 97% were correctly classified after hard limiting the output to an integer class. With the reduced raw data 100% of Irises were returned as the correct class with no further processing required. The difference in performance error was 10^{-9} less for the reduced data, producing a more accurate result.

Conducting an experiment with a 3 layer network with different numbers of hidden layer neurons also returns a similar result. With 10 neurons in the hidden layer, the maximum error in classification for raw data is 1, where the maximum for the processed is 10^{-8}. With 20 neurons, the raw data training error increases to 2 at the point where training reaches the best performance; processed data again has a maximum classification error of 10^{-8}. These results (displayed in Table V) show that the networks are much more receptive for classification if the data being classified can first be linearly separated.

Table V Comparison of training performance of non-reduced and reduced Fisher's Iris data in Feed-Forward Back-Propagation networks

Number of Layers	***Number of Neurons***	***Performance Error***	
		Original Dimension	*Reduced Dimension*
2	20	3	0
3	10	1	10^{-8}
3	20	2	10^{-8}

Given that the data can be separated in the lower dimension space with this method, it is now suited to simple binary classification; one perceptron could be used for each cluster whereby the output is 1 or 0 depending on its membership to that cluster. Using the high dimension, nonlinearly separable data, the perceptron training fails to converge and therefore cannot be used as a correct classifier. In Matlab, three perceptrons are trained with the raw data and the processed data, to a maximum of 1000 epochs each or earlier if the performance reaches the required value. Figure 39 shows the training performances with a comparison detailed in Table VI, with the Virginica and Versicolor sets in the high-dimension failing to achieve a suitable weight and bias value allowing for generalization. In the lower dimension, all sets are correctly trained, with results reflecting this (Figure 40). Presenting values to the higher-dimension-trained networks gave an error of 60% for the two conflicting sets, where the lower-dimension-trained networks had a 0% error rate.

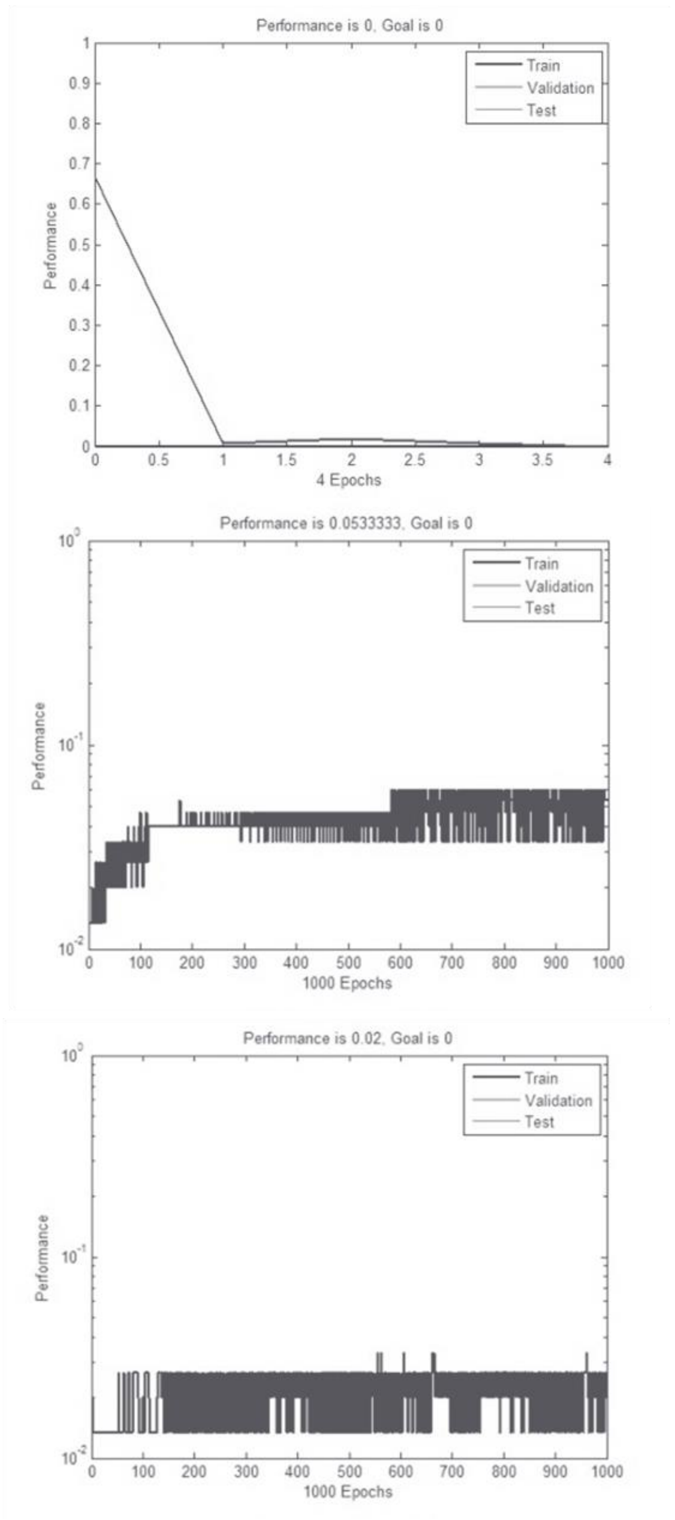


Figure 39 Results of training 3 perceptrons with Raw Fisher's Iris Data. Note the lower two results showing non-converging performance at maximum iterations, therefore not being linearly separable

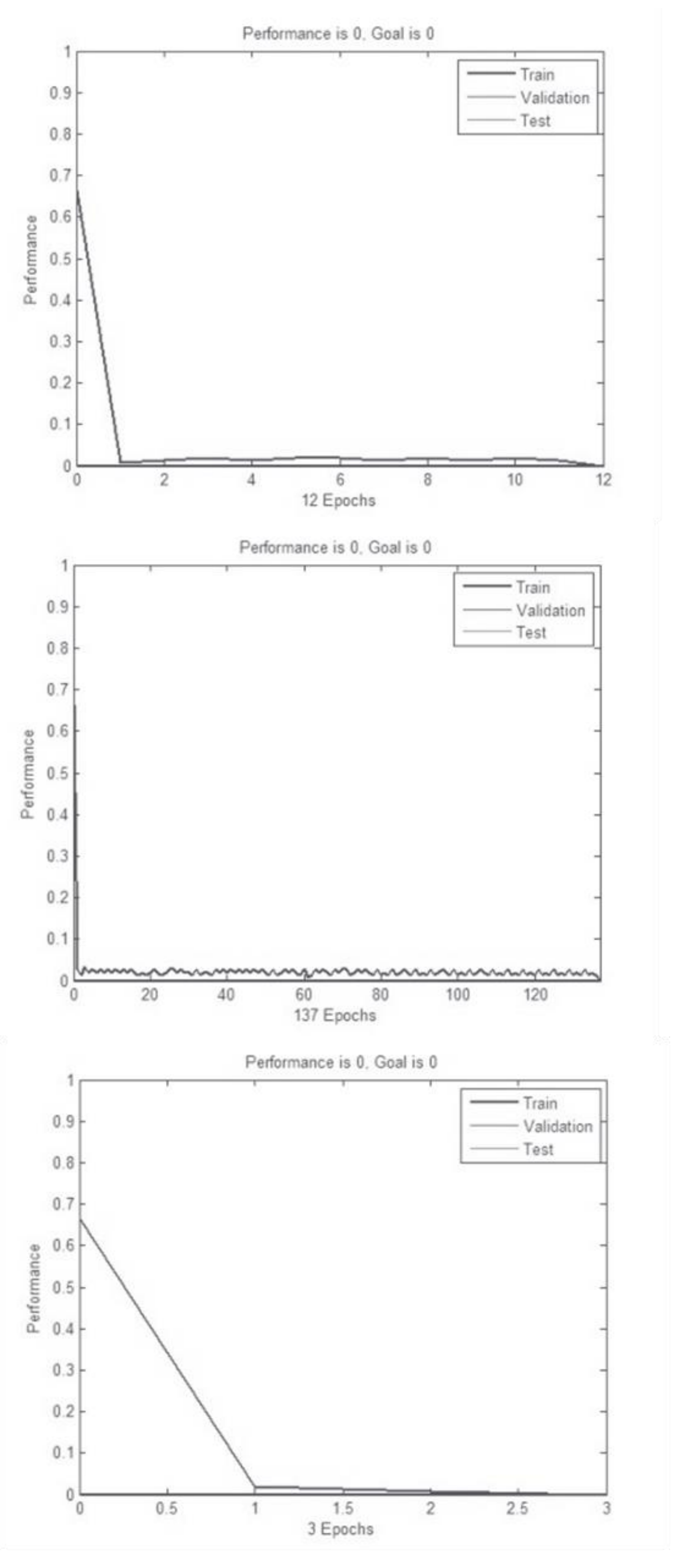


Figure 40 Results of training 3 perceptrons with Dimensionally Reduced Fisher's Iris Data. Note that now, the lower two results show converging performance before reaching the maximum number of iterations, thus indicating that all 3 classes are linearly separable

Table VI Further comparison of training performance of non-reduced and reduced Fisher's Iris data in the Perceptron

Iris	***Performance Error***		***Epochs***	
	Original Dimension	*Reduced Dimension*	*Original Dimension*	*Reduced Dimension*
Setosa	0	0	4	12
Virginica	0.0533	0	1000	137
Versicolor	0.02	0	1000	3

The dimension reduction and classification scheme is, in its initial offline training, a time consuming process for large data sets. With high-dimensional data consisting of an extensive amount of data points, the vector quantisation and linking steps can take copious amounts of computer operation cycles, yet it is in the reduction of projection error of the prototypes that we find the lengthiest process. Given a vast amount of data, linking the prototypes which generalise the component clusters results in an expansive graph which must be evaluated in every time-step during the minimisation of the projection error. When the projection space is of multiple orders of 10 larger than the initial data space (due to the "flattening" of the manifolds in the reduction method), the random projections which are to be relocated to minimise error can require that great distances be traversed in order to reach their optimum placement. In the above tests the 5 cluster artificial data took longest to optimise placements to a highly accurate degree (multiple experiments took 30-40 seconds each), yet once the prototypes of the training data are located, successive interpolation of unseen points can be done in real-time in a matter of cycles to a very high degree of accuracy (much quicker than with a *k*NN approach to classification). With this speedy interpolation being followed by the employment of a simple perceptron, the combined system is both fast and accurate in its classification of unseen data points from a nonlinear high dimension data set.

The only intended variable parameter in this system is the tolerable loss of the vector quantization step. Setting the value to close to 1 will result in fewer prototypes of the data and therefore faster projections, yet the trade-off is a lower accuracy result. With a lower tolerable loss (all examples here used 0.1) the reproduction maintains obvious shape characteristics of the clusters from the high dimension given the increased number of prototypes (the 3D artificial data shows that the "bowl" shape is inherent within the reduced projection). Ultimately it is at the discretion of the user of the method to decide on the accuracy of the topology reproduction given its trade-off with training speed.

The interpolation technique places the successive points in the lower dimension space within a few short iterations, and as such once the data's clusters have been prototyped it is possible to use the system in a real-time application to determine membership to a state. A problem may arise if the prototypes of the data begin to generalise only a small subset of a cluster. As the system evolves, more data may be present in a cluster which causes them to become sparser and as a result the prototypes don't generalise that cluster as well as with a more compact data set. Therefore during interpolation, closeness to the correct state can be reduced. Re-evaluating the system's prototypes periodically will overcome this issue, and a further enhancement could be achieved through ensuring that prototypes of the data also encompass boundaries of the clusters themselves.

At present, the nearest prototype to a data point determines the state to which that point belongs. It is supposed that if prototypes have a "strength" or "weighting" of belonging to a cluster, the closest prototype can be evaluated to see how likely it is that the new point belongs to it. If the distance to the next prototype is further, yet the strength of belonging to that cluster is greater, a trade-off mechanism can be evaluated to more adequately classify the data point to a cluster. It is in the adjusting of this neighbourhood function that the method will become a more useful tool in the supervised classification of data sets and the successive unsupervised interpolation of unseen data points. The next chapter details the enhancements made to the *k*-Nearest Neighbours classification method employed in part within the dimension reduction scheme. Where the manifold learning and classification here worked on a global scale, relating data points to each other according to curvilinear distances through high dimension manifolds, the *k*NN classification was only utilised for an indication of local topology and similarity determining on a localised scale. Both methods have merit when combined in the above way, as they serve to provide a "complete" picture of the multi-dimension dataset, however when the classification within Verity only concerns local rather than global relationships, an enhancement to the *k*NN method could prove more suitable based on the computational speedup over this manifold learning and classification.

5.4 Summary

Our Curvilinear Distance Analysis for Linear Classification (CDALC) method consistently provides results which fully satisfy the criteria needed for interpolation and classification of unseen data points. In the dimension reduction we accomplish two feats: production of a data set which retains its general topology from the high to the low dimension (with an allowable error) - thereby maintaining overall data set characteristics - and an increased usability of the set for successive operations with linear classification methods. Linking of clusters embedded within the high dimension manifold through our new approach of determining maximum within-cluster distance always ensures linear separation after projection.

Experiments on a number of widely applied and utilised data sets show the suitability of the dimension reduction scheme not only for visualisation of high dimension data but in its ability to linearly separate data that typically would be ineffectively operated on with standard machine learning and intelligent methods. Taking Fisher’s Iris Data as a primary example it was shown that the two not-linearly separable classes within the data are not adequately classified with multi-layered networks and simple binary classifiers. Processing the data with the dimension reduction scheme produced 3 separate classes with a 100% correct classification rate for all networks which previously produced unsatisfactory results with the raw, unprocessed data.

For intrinsically nonlinear datasets with more than 2 dimensions the CDALC process has been shown to be suitable for linear classification and visualisation, however attention is turned towards classification speedup and storage requirement reduction in the next chapter by dealing with localised similarity and optimisation of the classification procedure, without the need to generalise a data set through the above CDALC method which models the data according to calculated prototypes.

Chapter 6
Optimisation of the Classification Procedure with Instance Based Learning

The Dimension Reduction and Classification scheme is well suited to applications requiring linear separation of nonlinear data sets and visualisation of data classes typically unviewable in their high dimension form. However, the time consumed in the determining of similarity between close data points in the projected lower dimension space is a drawback when a real-time application is considered. The two schemes proposed and compared in this chapter utilise Bloom Filter-based storage to create two types of what we term “Bloom Memory”, where the training data is optimally compressed into a single, indexable array which is then used as a simple look-up table to determine which class subsequent unseen data belongs to. Both vary in their mechanisms to reach the same goal; however it can be shown that each has its own merits depending on operational circumstances and both can be seen to draw operational aspects from biological memory equivalents. They are employed to optimise the speed and storage space of the standard instance-based learning (IBL), *k*-Nearest Neighbours (*k*NN) approach to classification and are shown to work adequately in the classification application where the high dimension source data is intrinsically nonlinear.

6.1 Principle

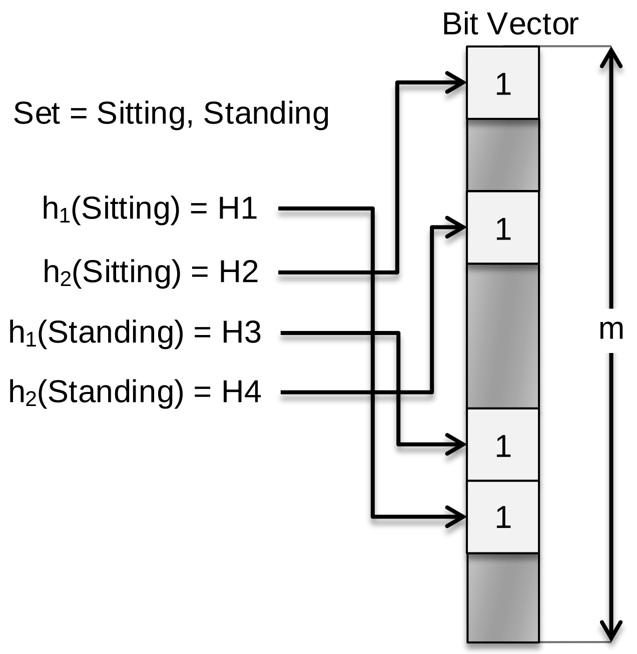


Figure 41 Bloom Filter principle with a set of two states as members and a filter length of *m*

The Bloom Filter is essentially a look-up table used to determine whether or not an element is present in a data set. A look-up operation is only of a complexity equal to the number of hash functions employed for encoding (h), which is vastly improved on that required in a search algorithm and is influenced by no other factors such as data set length. A set of input values are each submitted to a hash function to produce an integer value (“H”, in Figure 41) which identifies a location in the m-length bit vector to be set to a value of 1, thereby indicating the presence of a member of that set when checked during the look-up operation. Any number of input data – where each datum can be of any length – can be stored in the vector, provided that the correct number of hash functions is chosen to overcome the probability of false-positives, i.e. values set to 1 in the filter that signify the presence of an encoded value where in fact no encoding took place.

Using a Bloom Filter eliminates the need to store large sets in their raw form in a single structure, which, depending on the dimensionality of the input data, could in its uncompressed state take up extensive memory space which could otherwise be better utilised by an application. The limitation of the Bloom Filter if used unmodified for the classification operation is that if the query point is not already trained and present in the filter then there will be no matched-class output as a result. A solution would be to build on the structure of the Bloom Filter to exploit its speed in querying membership, whilst incorporating a method of searching possible neighbours of the query to determine whether these points are also encoded in the memory. In such a manner, the resultant “Bloom Memory” would be capable of exacting subsequent query checks after the training stage that would assess whether a query point belongs to the class of its neighbouring points if the actual query point is not present in the initial filter. The look-up speed would not be significantly reduced compared to the initial Bloom Filter form, as the checks for the neighbour point’s presence would also only require assessing the contents of h locations for each potential neighbour – therefore still exhibiting a speedup in comparison to a standard distance check of the initial data. It is the problem of determining the most appropriate way to address this variable neighbourhood search that is the focus of research documented in this chapter.

6.2 Hierarchical Bloom Memory (HBM)

The simplest approach to the neighbourhood checking problem is in having multiple Bloom Filters employed for multiple resolutions of encoded data, where the highest resolution is akin to querying the data point itself and the lowest resolution querying the maximum acceptable neighbourhood around the original data point. Figure 42 illustrates the concept of the hierarchical construction of the filter layers (L_T) in the proposed Bloom Memory. A query point is initially checked for membership at the highest resolution vector of ω_0. If there is an activated entry in the layer at this resolution level, then the class (C) to which the point is attributed is most likely going to be that of the same point already contained within this filter. Continuing the querying operation with the other resolution layers would in theory yield results to confirm this fact, as with each layer in the memory the neighbourhood increases and therefore includes more instances belonging to the respective classes. Conversely, if the initial layer returns no activated entries then the resolution is decreased and the relevant layers are checked for activation.

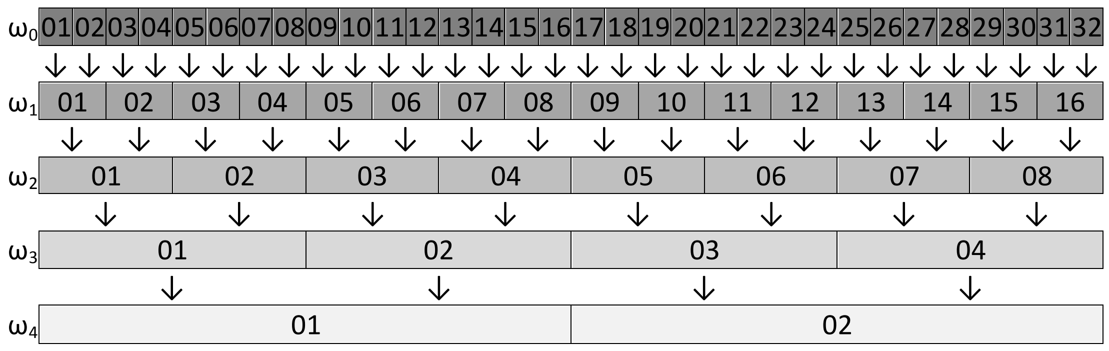

ω0 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
ω1 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
ω2 01 02 03 04 05 06 07 08
ω3 01 02 03 04
ω4 01 02

Figure 42 Hierarchical construction of Bloom Memory – the labels represent data points viewed as one dimension. The highest resolution of ω_0 is checked first, with subsequent resolutions being checked to ascertain likelihood of membership

This method of decreasing the resolution to determine point-presence is much like in the quantisation step of the Dimension Reduction Scheme developed in the preceding chapter, where an expanding neighbourhood radius considers all points it encompasses to determine the most likely membership class. As a variation on the *k*NN approach to classification, the speedup experienced by checking the Bloom Memory over the standard iteration through a data set for pairwise distance calculations significantly improves on real-time classification of data, especially in instances where the encoded data set is of multiple dimensions as in the Verity application. The speedup is significant during operation as the encoding process effectively pre-calculates the pairwise distances, thereby assigning neighbours to the same locations within the memory ready for the lookup operation.

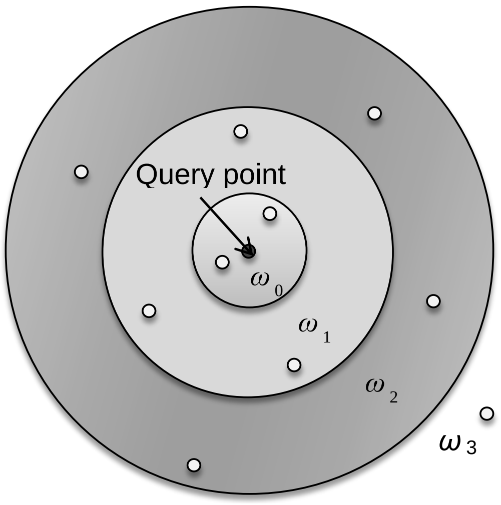


Figure 43 The expanding radius about a point, which can be considered its "neighbourhood"

Figure 43 shows the basic principle using the pre-set resolutions for neighbourhood radii – the central query point is here attributed to being in the same class as the two other points sharing resolution ω_0 with a higher probability than those in the decreasing resolutions of ω_1 to ω_3.

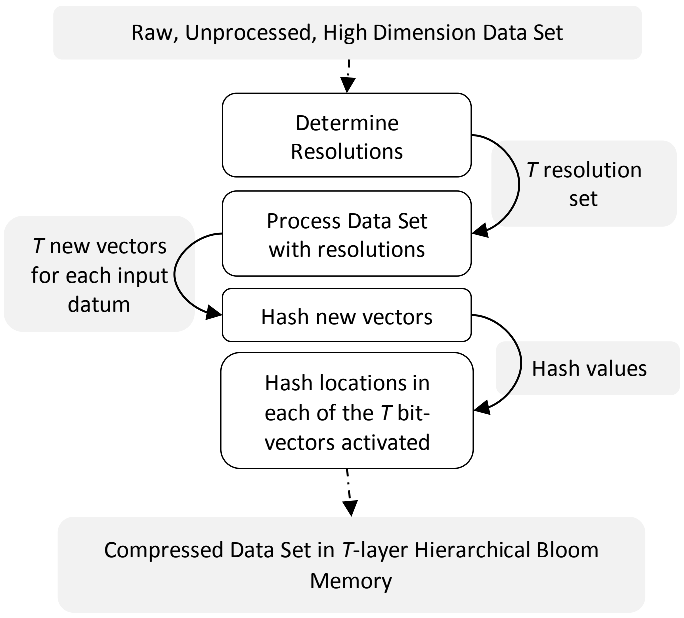

Raw, Unprocessed, High Dimension Data Set
Determine Resolutions
T resolution set
Process Data Set with resolutions
T new vectors for each input datum
Hash new vectors
Hash values
Hash locations in each of the T bit-vectors activated
Compressed Data Set in T-layer Hierarchical Bloom Memory

Figure 44 The encoding process for Hierarchical Bloom Memory

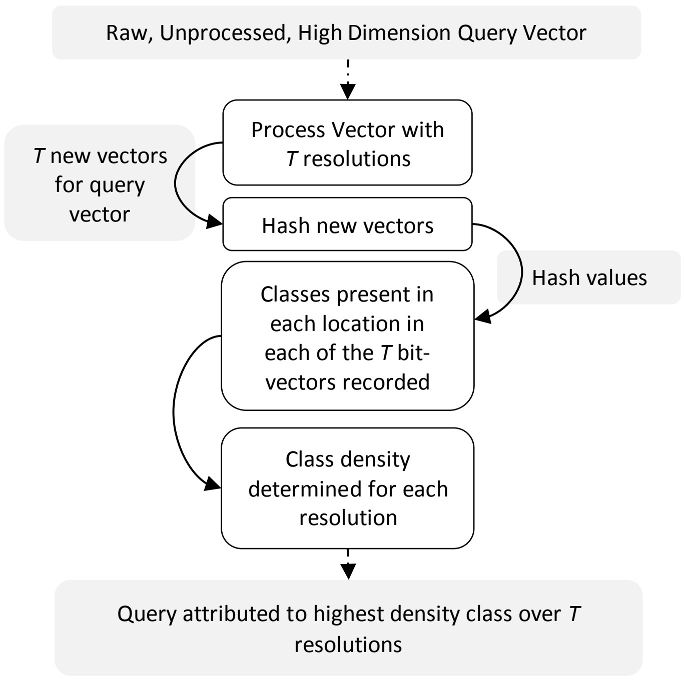

Raw, Unprocessed, High Dimension Query Vector
Process Vector with T resolutions
T new vectors for query vector
Hash new vectors
Hash values
Classes present in each location in each of the T bit-vectors recorded
Class density determined for each resolution
Query attributed to highest density class over T resolutions

Figure 45 The querying process for Hierarchical Bloom Memory

The encoding process for the Bloom Memory learns the input vectors and their associated states for subsequent point classification in the querying process. An input vector is pre-processed to ascertain which resolutions it will fall into, before these multiple resolution values are hashed and the resultant respective locations in the memory are activated with a value which represents that vector's class (Figure 44). Classification simply requires knowledge of which resolution the query point falls into in the memory, in which there are also some corresponding points previously encoded by the learning step. The multiple resolutions that the query point can belong to are first determined through the same formula as in the encoding scheme, before these resolution values are hashed and checked in the memory for activation according to the idea above. The presence of an active bit at the location specified by the hash function indicates that there is another point within the neighbourhood of the query point that was previously encoded in the memory. The classes of these points identified at the location are recorded, with the probability of membership determined according to the density of these recorded classes about the query point (Figure 45). As the resolution decreases (and the neighbourhood radius expands), the weighting of the densities of the classes decreases similarly, according to the theory that the further away a point is from a query the less likely it is that they are related.

6.2.1 Encoding

The first task to be completed before the populating of the memory is the decision of the base and power values which determine the resolutions encoded into the memory. These T resolutions (for example ω_0 to ω_3 as in Figure 43) are used to compute the area in which an unseen instance N at that resolution most likely belongs when it is presented to the filter, so after training all that is needed for classification are the initially used hash functions and the multiple resolutions that the filter is storing. A resolution base value b is used to encode each data point into the memory before subsequent resolutions are influenced by the chosen power-base value p. Generalising, where $\lfloor \cdot \rfloor$ is the floor operator:

$$\omega_t(N) = p^{t+1} \times \left\lfloor \frac{N}{\left(p^t \times b\right)} \right\rfloor \qquad 0 \le t < T \quad (75)$$

The choice of b is best influenced by the precision of the raw data. For example, data sets with points accurate to 10^{-3} would be best allocated to resolutions with a base of 10^{-3} – any higher may result in the misgrouping of non-similar points for the first layer of the memory, whilst any lower would make the resolution operations irrelevant in the first few layers. p is chosen such that the resolution operation "rounds-down" the value of the data point to that degree of accuracy as t increases. An example of the process (with single-dimension data points, for simplicity) is shown in Table VII; t increases to the total number of resolutions T desired to be stored in the memory – chosen either based on available storage space or required degree of point generalisation.

A standard Bloom Filter size, m, is affected by the minimum number of inserted elements, n, for a given false-positive rate P_{fp} - assuming an optimal number of hash functions h is chosen (according to $h = \frac{m}{n}\ln 2$) [153]:

$$m = -\frac{n\ln(P_{fp})}{(\ln 2)^2} \tag{76}$$

According to equation (76), the maximum size of the Hierarchical Bloom Memory is now also affected by the number of resolutions, T, chosen to represent the data; each resolution receives its own layer in the memory:

$$\mathrm{Mem}_{size} = -\frac{n\ln(P_{fp})T}{(\ln 2)^2} \tag{77}$$

Each decreasing resolution should, however, require a lesser-sized layer due to the lesser number of inserted elements – therefore the estimate for memory size in (77) should never theoretically be reached and acts solely as an indication of upper limit for the size of the memory. Table VII for example would require the insertion of 5 elements for resolution ω_0, where resolution ω_4 combines two input vectors into the same element and therefore requires only 4 element insertions.

Table VII Example resolution operation, with *b*=10^{-3} and *p*=10.

Raw Data (*N*)	ω_0	ω_1	ω_2	ω_3	ω_4
132.589	1 325 890	1 325 800	1 325 000	1 320 000	1 300 000
129.999	1 299 990	1 299 900	1 299 000	1 290 000	1 200 000
86.426	864 260	864 200	864 000	860 000	800 000
85.512	855 120	855 100	855 000	850 000	800 000
0.985	9850	9800	9000	0	0

When the parameters of the Bloom Memory have been defined, the encoding of each instance and its various resolution values into each layer (L) can take place. The raw data is typically of vector form, which is hashed as a whole as a sequence of integers with either a single, or multiple functions to produce h integers used as indexes for locations within the layers of the memory, as per Figure 42. Each memory layer corresponds to a resolution, with all buckets in a layer capable of holding a number of values equal to the number of classes, S, in the data set. The system works similarly to a counting Bloom Filter [115], by incrementing a "bucket" by 1 every time it is hit by one of the h hash function's outputs. The bucket hit count is accompanied by a label which indicates to which class, α, the hits belong (78), as illustrated by Figure 46.

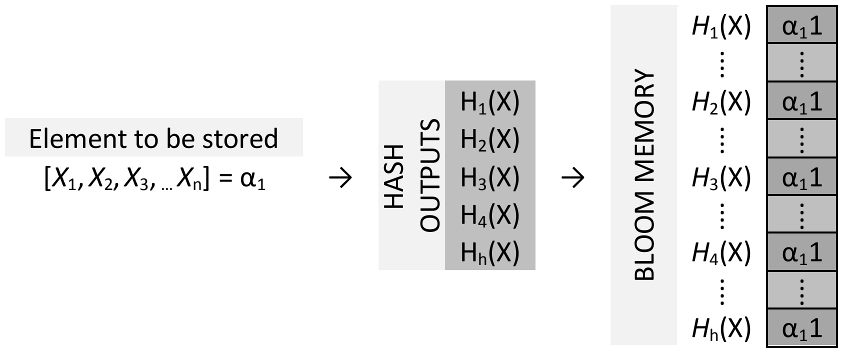


Figure 46 Insertion of an element into a layer in the memory, with the labelled "bucket" indicating the class to which the element belongs

Here L_t represents the resolution layer of the memory (to a maximum of T layers), with H_x equal to the one of the h locations given by hashing the element vector belonging to class α_s: which is also the state value (C) inserted in the memory at the same location.

$$L_t(H_x) = \left(L_t(H_x)+1, C\left(L_t(H_x)\right)=\alpha_s\right) \qquad \begin{matrix} 0 \le t < T \\ 0 \le x < h \end{matrix} \qquad (78)$$

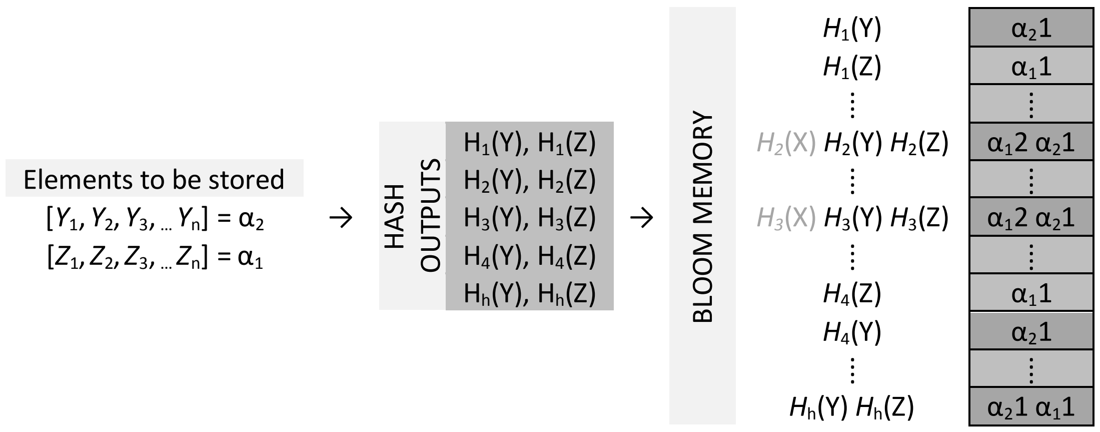


Figure 47 Insertion of 2 other elements into a layer in the memory, with one belonging to the same class as the first

If there is already a hit count present in the layer at the location indicated by a hash function, the class to which the count belongs is checked.

If the class differs from that of the element being inserted, a new count is begun adjacent to this in that bucket and the new class label associated with it – otherwise, the class's hit count increases (as shown in Figure 47, where an element is inserted in a location which already contains that class). The element is inserted into each layer at its various resolutions, such that the memory contains a record of it and its localised area ready for subsequent lookups.

6.2.2 Querying

The standard Bloom Filter implementation uses simple element look-ups to determine whether or not an element is present in the storage array. The query vector is hashed with the same hash functions used for the original data storage, and each hash-returned location in the bit-vector is checked for the presence of an active "1". If all locations return activations, there is a significant probability that the query vector has been seen by the Bloom Filter by an amount which factors in the false positive probability – that is the chance that all activations are not in fact the result of the one element having been seen but a number of elements that happened to hash to the same locations.

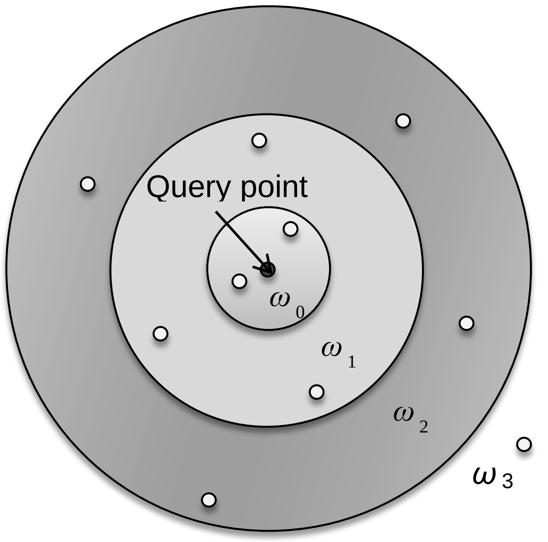
For the Bloom Memory, the condition which must be true to indicate the presence of the element (N) in the resolution layer t can be expressed as:

$$\sum_{1 \le x \le h} \left(L_t\left(H_x(N)\right) \mid C\left(L_t\left(H_x\right)\right) = \alpha_s, L_t\left(H_x(N)\right) \ge 1 \right) \ge h \qquad \begin{matrix} 0 \le t < T \\ 0 \le s < S \end{matrix} \qquad (79)$$

i.e. at each location in a layer – given by the hash of the data point – the total hit count for a given class must be at least 1. If non-zero values are present at each of the hash locations, the layer therefore possibly contains a record of the data point and the number of activations is recorded to provide an idea of how many of the same instances have been encoded to it. Given the counting-nature of the memory, the highest possible number of points that could have been seen is equal to the lowest number over all of the active locations for that data point:

$$activated_{\min} = \min_{0 \le x \le h} \left[L_t\left(H_x(N)\right) \mid C\left(L_t\left(H_x\right)\right) = \alpha_s \right] \qquad 0 \le s < S \qquad (80)$$

This number equates to the possible number of instances that match the query point that are present in that layer of the memory, therefore the number of matching points within that resolution. The membership query must propagate through all layers of the memory in order to enable the optimum decision as to which class in which resolution the point belongs to. In order to facilitate look-ups, the different resolutions of the query point are calculated and submitted to the layers, with the number of points present at each layer recorded. The aggregation of these results inform of the density of points for each class within the point's neighbourhood, up to the maximum resolution step offered by the memory. Figure 48 illustrates the principle of the results of a query; points between resolutions are the utilised values. Each circular representation of a resolution value is synonymous with a layer in the memory.



Resolution	Point Running Total	Points Located Between Resolutions
ω_0	2	2
ω_1	5	3
ω_2	9	4
ω_3	10	1

Figure 48 Example to explain the principle of point location within radii of a query point

The decision to classify the instant to a specific class is based on the density of points in the surrounding area of the query point, with the theory that a higher density in the immediate surrounding area would realistically be a better match for a point that was surrounded by a lower density. With the scheme structured as it is - allocating points on a resolution basis - the density of points in each resolution for that class have a varying influence on the query point's classification. If the class has a lower number of training points in a higher resolution over a class with a higher number of points in a lower resolution, the weighting of the resolutions determine which class wins out. The initial calculation required is termed the Layer Density, which determines the proportion of points allocated to a class at a single layer level in all classes. The number of points in a previous layer is discounted, as this has already been considered by a previous layer calculation and so is subtracted to give the number of points solely in this resolution.

$$\mathrm{LD}_t(\alpha_s) = \frac{L_t(\alpha_s) - \sum_{t-1 \geq i \geq 0} L_i(\alpha_s)}{\sum\limits_{0 \leq s < S} L_t(\alpha_s)} \qquad 0 \leq s < S \qquad (81)$$

The Weighted Layer Density accounts for how much of an influence the resolution is on the classification of the query point. Points in a lower resolution have a lesser influence as they may be present at the very far end of the resolution's range, therefore have little bearing on the query point when compared to points in higher resolutions and thus closer to the data point. The weighting value is taken from the resolution calculation, as each resolution step decreases the accuracy of the query point by the power-base value p, therefore it is reasonable to assume the influence of any points in this region is also reduced by such a factor.

$$\mathrm{WLD}_t(\alpha_s) = \mathrm{LD}_t(\alpha_s) \times p^{-t} \qquad \begin{matrix} 0 \leq t < T \\ 0 \leq s < S \end{matrix} \qquad (82)$$

The Class Weighted Density is the sum over all layers' calculated densities within a single class.

$$\mathrm{CWD}(\alpha_s) = \sum_{0 \leq t < T} \mathrm{WLD}_t(\alpha_s) \qquad 0 \leq s < S \qquad (83)$$

To use the calculated values in a probability scheme, thereby associating the classification with a probability of occurrence, the values of the CWD for each class are calculated relative to those of the other classes to give the Class Probability.

$$\mathrm{CP}(\alpha_s) = \frac{\mathrm{CWD}(\alpha_s)}{\sum\limits_{0 \leq s < S} \mathrm{CWD}(\alpha_s)} \qquad 0 \leq s < S \qquad (84)$$

The query point is then classified as belonging to the class which has the highest CP over all classes, as this class is most likely to have seen the query point due to the density of similar points over all layers. It may be the case that the class has the exact-same point in its highest layer, or the lower layers have a much larger density of similar points than those of other classes.

6.2.3 Experimental Results

A 20000 vector data set [154], consisting of 16 dimensions per vector and describing written characters belonging to the standard 26-character English alphabet was used to validate the effectiveness of the above methodology. The set was split into a training and classification set, with just under 15000 vectors being inserted into the memory during training and the remainder being used for validation. Fisher's Iris Data was also used as a secondary training set due to its commonplace use for validation in *k*NN-alternative schemes, with just under 25% of the data set's vectors being used for classification purposes (i.e. not seen during training). The resolutions used for testing had a base value b of 2^{-1} and a power value p of 2, due to the fact that the raw values were already integers and were no larger than 15 in any dimension – any lower base values or higher power values would have resulted in different resolutions outputting the same values. Comparison with the *k*NN classifier being sought to optimise returned the results displayed in Table VIII. Evidently (and as expected), the true positive success rate is suitably high for the standard *k*NN approach to classification, with the single parameter of an adjustable *k*. The highest correct classification percentage yielded for the letter recognition set resulted from classifying according to the single closest neighbour to the query point in this data set, at a time cost of over 2 and a half minutes. Similarly for the Iris data the results were all as expected, with 100% classification across all variations.

Table VIII Experimental results for Hierarchical Bloom Memory

Data Set		*k*NN Classification			Hierarchical Bloom Memory		
		k=1	*k*=3	*k*=5	p=2, b=2^{-1}, ω=2	p=2, b=2^{-1}, ω=5	p=2, b=2^{-1}, ω=10
Letter Recognition	% Correct	94.81	94.81	94.77	64.54	66.56	66.56
	Time (ms)	162 073	162 631	163 042	858	2839	6784
Iris	% Correct	100	100	100	100	100	100
	Time (ms)	82	274	314	5	6	26

With the Hierarchical Bloom Memory, whilst the highest correct classification percentage is less for the letter data, the time cost associated with such a result is at most just under 3 seconds (66% correct). If the reduction by 2% in true positives is acceptable (to 64% correct), the time cost further reduces to just less than 1 second (858ms). The main advantage of this memory variant for the Iris data, as can be seen above, is the incredible reduction in time cost associated with the classification. For no loss in the recognition rate, classification can be completed in almost 1/20th of the time when compared to the standard *k*NN approach. As an aside to the results in the table above, where the test determined how well an untrained data point was classified: for direct classification of the input training set i.e. a check to determine how well the memory has recall ability, the time cost is approximately 6 seconds, with a correct classification rate of 97.3% for the letter recognition data and less than 0.01 second and 100% for the Iris data. Compare this with a *k*NN approach applied to the same letter recognition training data – with each value in the entire set being compared pairwise for distances to each other to determine classification – and it is evident that the Bloom Memory can recall stored data much quicker than utilising a *k*NN method which returns 100% correct classifications in 7.5 minutes when using a value of k=3.

The Hierarchical Bloom Memory takes into account the density and weighting of the data points in the data set for each of the classes, providing a classification result in the form of probabilities of membership to each class. Density-based clustering has proven successful in identification of outliers as it considers sparse and dense regions within the data yet it has not been significantly employed for the purposes of classification based on the same principle, a fact observed by Plant *et al.* [155]. The authors introduce a classification scheme based on the local density of a query point, assigning it to the class in which it most appropriately fits according to the set of within-class *k*NN and their densities. Similar to the implementation of the Hierarchical Bloom Memory, the class determining process relies on a number of calculated factors based on the density of classes about a point, first identifying the mean distance between said point and the *k*NN belonging to each class. This initial value of “Direct Density” is said to enhance the pure *k*NN approach to classification as it has no majority voting and therefore objects belonging to rarer classes have the ability for correct classification based on their density in a data set.

The secondary decision value determines the degree to which a point can be considered an outlier of a class, by first identifying the density of the class itself (average within-class distance) and then dividing the Direct Density by this “Indirect Density”, to arrive at a value approximating 1 if the point is situated within a distinct class boundary, and a value higher as the point is located further beyond the majority and can be considered as an outlier. Assigning the point according solely to the outlier factor is shown to be ineffective where a point lies in similarly dense class regions, but combination with the Direct Density to assign it to the class with a low outlier factor and higher class frequency in the local neighbourhood proves more successful as it considers both the global and local regions about a point.

This density-based method of class identification is somewhat more useful in sets with interweaved classes, where the boundaries are indeterminable and a data point isn’t necessarily going to belong to the class with a data point in the immediate vicinity, as in those sets employed for experimentation in the above authored work and this research contained here. In the case of non-linearly separable data, quite often the classes are sparse and overlap so these similar methods of classification can prove not only beneficial in their result form (outlier factors and probabilities in the respective cases), but in their speed of processing large data sets and classification effectiveness when compared with the standard *k*NN approach.

Whilst the Hierarchical Bloom Memory is effective in its operation in terms of speed over a standard search approach as evidenced by the results above, the storage space required is variable depending on results desired and can perhaps in an extreme circumstance increase on the memory utilised in storage of the raw data for a *k*NN approach. The reasoning - whilst the performance speed will always be increased over the iterative process used by standard searches provided that the number of hash functions utilised is less than the size of the data set - is that classification performance significantly relies on the number of resolutions encoded within the memory, where a higher number of stored data resolutions results in greater accuracy, yet more layers. As in the letter recognition set used above, at the lowest level (highest resolution) it may be that the stored values aren’t falling within the same data range as the query values, thereby resulting in a resolution decrease to where queries fall in multiple classes according to density in that data range and performance correctness reduces (where in a *k*NN approach there would be no such experience). A mid-resolution spanning both of these would perhaps increase in the classification accuracy at a cost of increased storage within the memory; therefore it is a trade-off between space and performance in this set-resolution approach.

A variable resolution method with the ability to encode the pure data values and then assess a variable neighbourhood during the query process would increase the performance ability, without compromise of storage space and with a visible speedup in encoding. Adapting the memory scheme to utilise more “intelligent” hashes generated by Locality Sensitive Hash (LSH) functions, as first described in [110], affords the encoding process the ability to act solely on raw values without need to pre-process to provide a multiple resolution data set.

Locality Sensitive Hash functions are a family of hash functions which speed up the nearest neighbour search by hashing vectors considered as neighbours to the same locations with a higher probability than vectors further apart. In a *k*NN application, LSH functions are employed to encode data vectors according to this principle into a data structure which at each location will then contain neighbouring vectors, therefore a query operation will return multiple vectors encoded at the hash location which then can be acted upon with standard distance metrics to obtain those *k*-Nearest Neighbours. Rather than having to iterate through an entire data set and suffer from substantial computation in neighbour-finding, the LSH approach speeds up the operation to return only those vectors with a possibility of being neighbours on which the more detailed operations can occur to find the definitive neighbouring vectors.

Since their inception, multiple hash table-based applications have been developed to exploit the LSH properties to optimise the *k*NN algorithm [156 , 157], yet there has not been documented a technique developed in order to produce a classifier capable of both determining the most likely state for a query and its likelihood of membership using a similar approach to the density-based classification detailed above. LSH implementations require the storage of the original vectors for detailed classification operations and therefore data structures quite often exist alongside the original vector storage and act solely as look-up tables, thereby allowing for only computational speedup and not storage optimisation.

For the application we deal with here and for general optimisation of the *k*NN IBL method, we remove the need to process a data set into different resolutions for storage within a Bloom Filter and replace the standard hash functions with LSH functions which are able to maintain the neighbourhood of similar vectors/points from the original data set when they encode the vector to memory. This then removes the need for multiple resolution layers within the memory, as a check for similar points within a radius requires only a search of neighbouring buckets to the query instead of multiple checks to the layers within the previous memory implementation’s hierarchy. The result is a more compact memory, consisting of much less entries over the HBM method and an increased operability due to the variable neighbourhood – the radius to be assessed in order to determine a possible/likely class for a query point can be adjusted to achieve the desired functionality and classification accuracy.

6.3 Localised Bloom Memory (LBM)

So called due to its ability to retain the proximity relationship between raw points, neighbours and their encoded counterparts, the Localised Bloom Memory modifies the previous Hierarchical Bloom Memory mostly through use of the Locality Sensitive Hash Functions and the initial data processing. Due to the nature of an LSH, there are certain parameters that must be defined in order to make the hashes usable in the memory application. A simple LSH function can be described by:

$$LSH_h = \left\lfloor \frac{\vec{z}_h \cdot \vec{v} + b}{\omega} \right\rfloor \tag{85}$$

Where v is the vector to be hashed and Z is a family of random vectors, chosen at random from a Gaussian distribution, for example $N[0,1]$. Another random value b in the range $[0,\omega)$ is then added to the scalar projection which is then quantised by ω, which is the width of the bin in which a data point may fall into. $\lfloor \cdot \rfloor$ is the floor operator. Operationally, ω is akin to a resolution used in the previous Hierarchical Bloom Memory, as it determines how close two vectors must be in order for them to be hashed to the same bucket. Ideally, similar vectors would be located in neighbouring buckets, so the accuracy of the hash in recreating a distance relationship from the d dimensional space to the storage space is dependent on this value of ω. A larger ω will result in more vectors hashed to the same points. Figure 49 graphically explains an LSH function, with the point v here being hashed to the same bucket (9 here) as the last data point as its value of b falls within the bucket bounds of ω on the projection vector of z_h.

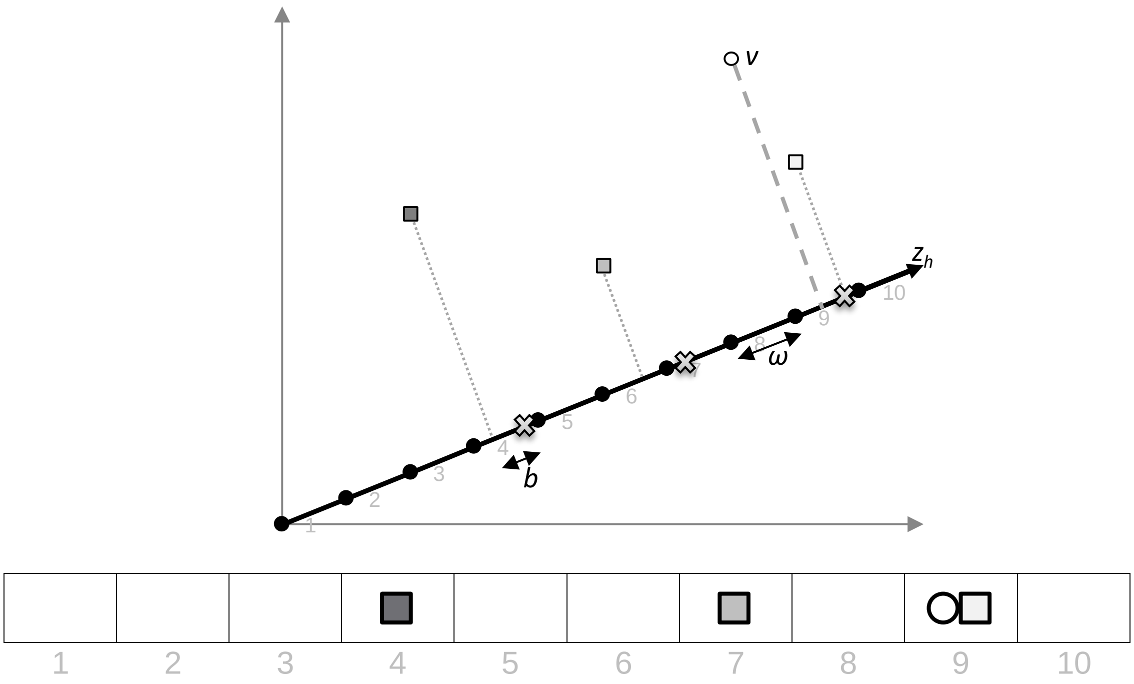


Figure 49 Geometry of Locality Sensitive Hash functions and resultant storage in a hash table

If the data vectors to be trained are accurate to a value of 0.1, it would be advisable to set this value as ω, where similarly integer input vectors would benefit from having the value of ω set to 1 to ensure neighbouring vectors are only located 1 bucket away after the floor operation. This value directly affects the size of the memory, m, if the family of random vectors Z adhere to the suggested values ($N[0,1]$), where $\lceil \cdot \rceil$ is the ceiling operator:

$$m <= \left\lceil \frac{\sum_{0 \le i < D} \max_{0 \le x < n} \left[\left(\max [Z] \right) \cdot v_x \left(d_i \right) \right]}{\omega} \right\rceil \tag{86}$$

The value of m is therefore at most equal to the sum of the projected highest value of each dimension, divided by the bucket width, due to the random vector family having a maximum value of 1 in each dimension – thus the maximum dot product is determined by the input vector alone.

6.3.1 Encoding

The size, h, of the family of random vectors (Z) is assigned according to accuracy of projection and in order to maximise the likelihood of neighbouring data vectors to fall into the same bucket. As shown in Figure 49, the single projection vector in this instance places two points (including the query) in the same bucket whereas another projection vector may not. Ideally, the number of random vectors should be large enough to ensure neighbouring projections whilst increasing the dataset compression amount. The encoding process for each vector in the data set can begin after the family of random vectors are defined, and follows the same operation as the previous memory but utilises the LSH function of equation (85) instead. The encoded values are placed into the memory associated with the respective hash functions, where each vector (V) is represented by a "hit" at a "bucket" much like the previous memory except this time the label of the vector (its index i, i.e. position in the training set) indicates its presence at that location (87).

$$memory_x\left(\left\lfloor \frac{\vec{z}_x \cdot \vec{v}_i + b_x}{\omega} \right\rfloor\right) = i \qquad \begin{matrix} 0 \le i < n \\ 0 \le x < h \end{matrix} \tag{87}$$

The memory structure is as the previous would be but with a single memory layer per hash function. From a visual perspective, the result after training should show activated buckets now appearing in clusters due to the locality properties of the hash functions maintaining proximities of similar points from the high dimension to the lower dimension. Figure 50 illustrates an example layout of the Localised Bloom Memory, where it can be seen that a higher concentration of points hash to the same "central" values of a cluster. On the two projection lines of Z_1 and Z_2 it can clearly be seen that data points do hash to different locations, yet both hash functions return memory arrays in which points considered neighbours are located next to each other. The more hash functions used, the more likely it will be that neighbouring points in the original data space hash to more neighbour locations in each array, thereby increasing certainty of neighbour relationships between vectors. It can be seen that our example vector V_1 with hash function Z_1 is placed in a location with V_3 and V_5, despite being either side of the projection line and therefore not classifiable as a neighbour. However, with Z_2 we find a more reliable projection where V_1 is placed in the same location as V_{10} which is in the immediate vicinity of our example vector.

Examination of the buckets either side of the location of V_1 returns again values considered more as neighbours, and it is in this property that we find the exploitable advantage over the Hierarchical approach as we can identify likely neighbours by expanding our bucket search rather than having to encode separate vectors for every resolution.

Unlike the Hierarchical Bloom Memory (Figure 46 and Figure 47), the bucket positions in the Localised variant contain indexes of encoded vectors rather than a “count” of hits for a state. This could theoretically result in a bucket holding a record of every single vector in the training set, however provided the number of LSH functions is less than the dimensionality of the training data there is always going to be an element of compression in this memory over the storage of the raw vectors. Comparatively with the HBM approach, the LBM implementation also improves on the storage as it stores only h rather than $h \times T$ values per vector in a single table.

As a consequence of the storage of indexes rather than vectors, there must be an index catalogue which indicates to which class each of those stored indexes belong (Figure 51). The indexes of the vectors are stored in ranges with a corresponding “class” association - this removes the need to store the data vectors themselves as in a traditional LSH Bloom Filter, where each of the vectors are also stored for recollection [112], as the actual values of the vectors and their distance to the query point are not used to determine the class of the query point but only the density of those classes surrounding it within a given radius.

Element to be stored

$[X_1, X_2, X_3, \ldots X_n] = V_1$

$\downarrow$

HASH OUTPUTS

$h_1 = 07$

$h_2 = 09$

$\downarrow$

	01	02	03	04	05	06	07	08	09	10	11	12	13
h_1	-	-	-	V_9	V_2	V_4 V_7	V_5 V_3 V_1	V_{10}	V_8	V_6	-	-	-
	01	02	03	04	05	06	07	08	09	10	11	12	13
h_2	-	-	V_2	V_7 V_9 V_5	V_3 V_8	V_6	-	V_4	V_1 V_{10}	-	-	-	-

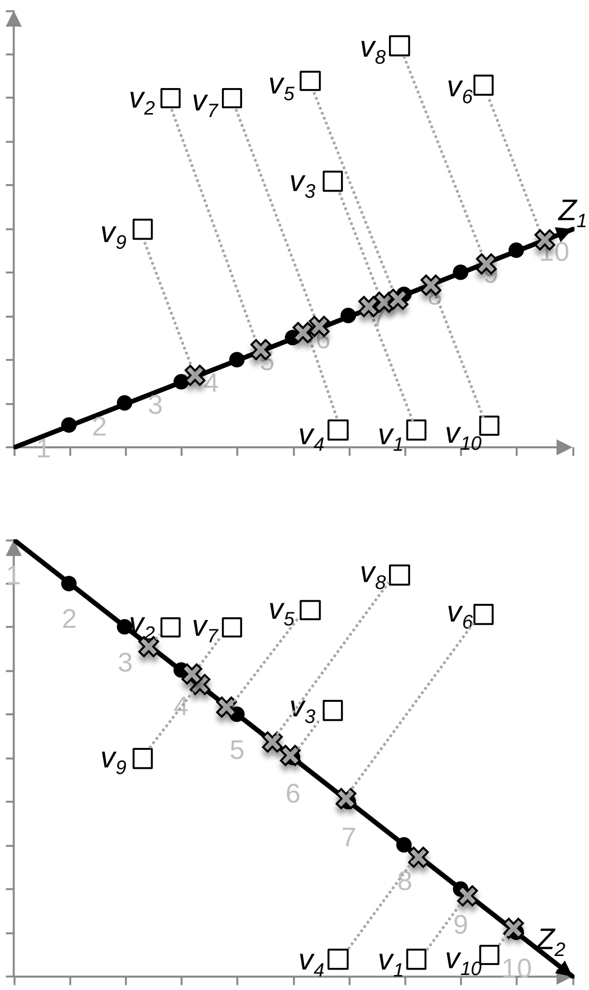


Figure 50 Construction of Localised Bloom Memory with one memory array used per hash function and a 2D representation of the LSH functions (here there are 2 hash functions, Z_1 and Z_2 producing hash values h_1 and h_2)

$[V_1, V_n(\alpha_1)]$	$[V_n(\alpha_1)+1, V_n(\alpha_2)]$	$[V_n(\alpha_2)+1, V_n(\alpha_3)]$	$[V_n(\alpha_3)+1, V_n(\alpha_4)]$	$[V_n(\alpha_4)+1, V_N]$
α_1	α_2	α_3	α_4	α_5

Figure 51 Construction of the index catalogue that accompanies the main Localised Bloom Memory structure for an example set of 5 classes

The first step is to cluster the input vector data $\left[V_1,V_N\right]$ into classes (α_s), assigning each member (V_x) a consecutive index value such that the entire set is ordered and in distinct classes, as in Figure 51 where a member range is equal to the next index after the last class member, $\left[V_n\left(\alpha_{s-1}\right)+1,V_n\left(\alpha_s\right)\right]$. The catalogue is then constructed such that each of these ranges is assigned the class to which it represents, so that a simple check of the Bloom Memory returning an index can then return its member class by a lookup of the range into which that index falls in the catalogue.

The storage of additional, unseen vectors after training is also very simple in comparison to other methods, as just the index of the new vector, V_i, is determined according to its class α_s in comparison to the current catalogue, and the memory at the required hash locations is updated as with any other vector encoded during training (88).

$$\begin{aligned} i &= \left|\alpha_s\right|+1 \\ V_n &= V_{n+1} \end{aligned} \qquad i+1>n>N \tag{88}$$

Once a new vector is to be added to the memory after training, the class to which it belongs $\left(\alpha_s\right)$ will increase its membership number by one. This will result in the index of the new vector being additional to the last index present for that class, thereby increasing that class' range: $V_n\left(\alpha_s\right)+1$. Subsequently, all other indexes for the other classes in the catalogue also increase their values by 1 to accommodate the increase in stored vectors. For instance, a new vector which is associated with class α_2 in Figure 51 is added to that range so the new range effectively becomes $\left[V_n\left(\alpha_1\right)+1,V_n\left(\alpha_2\right)+1\right]$. Subsequently, all index points in the memory greater than $V_n\left(\alpha_2\right)+1$ now increase by 1 to make space for the new vector in class $\left(\alpha_s\right)$, which is inserted at the relevant hash locations in the respective memory. Then the catalogue index ranges above the inserted point simply increase by one also and thus the memory has expanded to include the new value. Subtraction of a data point from the memory, perhaps due to irrelevance at some point in the future, simply reverses the procedure.

6.3.2 Querying

Again the retrieval method for this form of memory differs from the typical Bloom Filter, yet the operation is akin to the standard LSH recall procedure with “bucket” checking. There is no chance of collisions between data vectors because each vector has its own record in separate hash function memory layers, so once located at the hash locations there is 100% certainty that a point is encoded within the memory at the same location. This, however, is the variable element of this system: the size of each location. Neighbouring points are required to be encoded to neighbouring locations, thus the resolution decides what is considered a “neighbouring point” in the context of the training set. To query a data point, what is essentially required is to determine how many points lie within its neighbourhood and identify which class has the highest density and therefore probability to take ownership of the data point.

Due to the nature of the LSH functions, the radius which identifies as the neighbourhood can be variable unlike in the previous Hierarchical variation with its “set” resolutions. This is a significant advantage of this memory in the variability of the querying operation – once the data is stored, it can be operated on according to the application i.e. a wider radius to consider will possibly take into account an entire class in one operation, whereas a smaller radius will decide on the query point class based on immediate density. Thus, the querying operation first requires consideration of the radius step (r_{step}) and the maximum radius ($r_{\max}$) in order to determine whether the query vector V_Q or a neighbour is present (T_x is 1 or 0 depending on presence or not, respectively) in the hash function layer, $memory_x$. J here is the set of all indexes seen at all locations queried, with the presence of the query or a neighbour indicated if a member of the set is seen over all h hash function layers (89).

$$J = \left\{ memory_x \left(\left\lfloor \frac{\vec{z}_x \cdot \overrightarrow{V_Q} + b_x}{\omega} \right\rfloor \pm r \right) \right\} \qquad \begin{array}{c} 0 < x \le h \\ 0 \le r \le \dfrac{r_{\max}}{r_{\text{step}}} \end{array} \qquad (89)$$

$$V_Q \in memory_x \text{ if } \sum_{1 \le x \le h} T_x \ge h \qquad T_x = \begin{cases} 1, \; memory_x\left(\left\lfloor \dfrac{\vec{z}_x \cdot \overrightarrow{V_Q} + b_x}{\omega} \right\rfloor \pm r\right) = i \\ 0, \; memory_x\left(\left\lfloor \dfrac{\vec{z}_x \cdot \overrightarrow{V_Q} + b_x}{\omega} \right\rfloor \pm r\right) \ne i \end{cases} \quad (90)$$
$$i \in J$$
$$0 \le r \le \frac{r_{\max}}{r_{\text{step}}}$$

The radius search of bucket locations begins at 1 radius step, which is essentially the direct lookup of the query point's hash locations if the step/resolution is 1. If however the radius step is greater, the number of hash buckets considered for 1 radius is larger and thus a larger area about the query point (as in Figure 48) at each radius is considered. As before, each of the hash locations examined in the query process must contain a record of the same index point in order to be considered a storage location of a single point (90). With the expanding radius, however, the decision to state that a point is located in the neighbourhood of the query point can only be made once the maximum radius has been reached and all seen points are examined. Thus an indexed vector is only considered a neighbour of the query point if it is seen h times throughout the neighbourhood check. The radius at which the point is seen is determined to be equal to the maximum radius step at which one of the hashes identified it, and it is at this point that the index is checked against the catalogue and its class is recorded as being at this radius.

As with the hierarchical implementation, points located within the same immediate locality (1 resolution step) as the query point have an influence of 1 on the location of the query point. As each radius, r, is to be one more than the previous, then the influence factor is ½ that of the previous: as in theory there is twice the radius into which a point may fall around the query point. Within each radius, there will be multiple steps whose buckets will require checking and summing to give the total points within a radius. Once the total number of points at each radius for a class has been found ($r(\alpha_s)$), the scheme progresses like the Hierarchical Bloom Memory in order to ascertain a density value for the class. The Layer Density of (81) becomes the Radius Density:

$$\mathrm{RD}_r(\alpha_s) = \frac{r(\alpha_s)}{\sum\limits_{0 \le s < S} r(\alpha_s)} \qquad \begin{array}{c} 0 \le r < \dfrac{r_{\max}}{r_{\text{step}}} \\ 0 \le s < S \end{array} \tag{91}$$

The weighting of the density is then taken into account:

$$\mathrm{WRD}_r(\alpha_s) = \mathrm{RD}_r(\alpha_s) \times r^{-1} \qquad \begin{array}{c} 0 \le r < \dfrac{r_{\max}}{r_{\text{step}}} \\ 0 \le s < S \end{array} \tag{92}$$

At which point the Class Weighted Density and the Class Probabilities can be calculated, with the above new terms.

$$\mathrm{CWD}(\alpha_s) = \sum_{0 \le r < \frac{r_{\max}}{r_{\text{step}}}} \mathrm{WRD}_r(\alpha_s) \qquad 0 \le s < S \tag{93}$$

$$\mathrm{CP}(\alpha_s) = \frac{\mathrm{CWD}(\alpha_s)}{\sum\limits_{0 \le s < S} \mathrm{CWD}(\alpha_s)} \qquad 0 \le s < S \tag{94}$$

The query point is then classified as belonging to the class which has the highest CP over all classes, as this class is most likely to have seen the query point due to the density of similar points over all radii.

6.3.3 Experimental Results

Table IX Experimental results for Localised Bloom Memory versus *k*NN and the standard LSH Bloom FIlter

Data Set		***k*NN Classification**			**Localised Bloom Memory** LSH Bloom Filter		
		***k*=1**	***k*=3**	***k*=5**	**ω=0.001, r_{step}=5, $r_{\max}$=4000**	**ω=0.01, r_{step}=1, $r_{\max}$=450**	**ω=0.01, r_{step}=1, $r_{\max}$=400**
Letter Recognition	**% Correct**	94.81	94.81	94.77	80.5 80.17	85.03 81.27	74.71 73.96
	Time (ms)	162 073	162 631	163 042	99 309 99 000	65 302 64 295	38 260 38 200
					ω=0.01, r_{step}=1, $r_{\max}$=50	**ω=0.01, r_{step}=1, $r_{\max}$=40**	**ω=0.1, r_{step}=1, $r_{\max}$=5**
Iris	**% Correct**	100	100	100	100 100	100 100	100 100
	Time (ms)	82	274	314	7 7	5 5	4 4

The same data set as used before is submitted to the Localised Bloom Memory in order to validate the effectiveness of the above, modified methodology. The set was again split into training and classification sets, with none of the classification set being directly encoded into the memory before being presented for testing. The resolution ω, radius step r_{step} and maximum radius $r_{\max}$ are all variable, and the results of multiple combinations can be seen in Table IX. For these experiments, the hash number h was set at 10.

A correct classification is considered to be when the highest percentage of likelihood belongs to the class that matches the actual class of the query point. In reality, the output of the scheme is a series of probabilities that when combined further with other methods, i.e. Hidden Markov Models for time series modelling within Verity, will enable the correct state to be chosen based on more than just the classification percentage.

For the new LBM, the most accurate set of variables was found to achieve a correct classification rate of 85.03% with the Letter Recognition data. With Fishers' Iris Data, consistent correct classification rates of 100% for unseen data were returned, again at a fraction of the time required to compute values for the *k*NN operation. The Iris results show that the density calculation methodology with locality sensitive hashing can improve upon the standard nearest-neighbour approach, as more properties of the data set are considered than solely the proximity of data points in a significantly shorter time. To possibly improve on the LBM performance with the Letter Recognition data, the number of hash functions was increased to 30, with resolution, step and maximum radius set at 0.01, 10 and 450 respectively. Whilst the storage requirement increases over a raw vector and *k*NN approach at this hash number, the performance for classification increases to 88% in a time of 100419ms – another speedup in classification over *k*NN for a drop of only 7% accuracy for a large data set.

Whilst carrying out the experiments with the new Localised Bloom Memory, a test was run at the same time to assess the accuracy when compared with the standard LSH method. As LSH functions store data in a compressed format yet retain the proximity values of the high dimension, typically alongside the raw vector data a hash table can be maintained containing the compressed, indexed data. This enables much faster searching through all vectors to determine the closest as in a *k*NN approach: the values within the "radius" of a query point are the only vectors to have their Euclidean Distance to the point determined, rather than the entire data set. The speed improvement over a *k*NN approach does rely on a trade-off with the maximum neighbourhood radius, however, as can be seen in the earlier tests. The difference in accuracy between the new density calculation method and the nearest-neighbour approach with the LSH method is very small (all tests with the Letter Recognition and Iris data returned differences of just ~1% in favour of the new method), but with improved accuracy seen as the variable parameters are more adequately identified. In many cases, the nearest neighbour to the query point within a specified radius would not indicate the correct class, whereas the consideration of multiple points and their density in relation to the query point returned the correct classification.

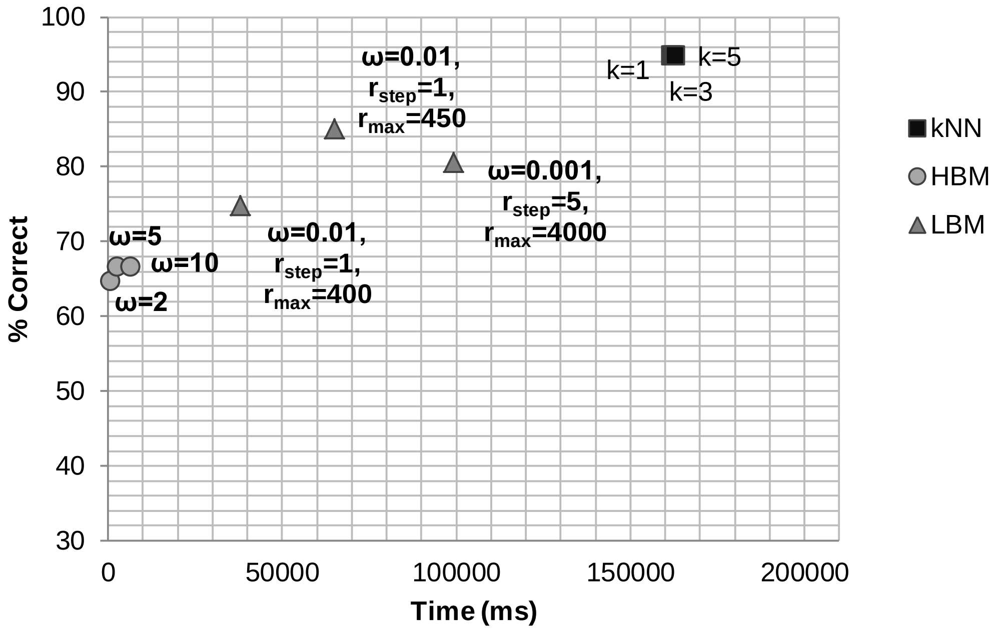


Figure 52 Letter Recognition Data Set results with both Hierarchical Bloom Memory and Localised Bloom Memory variants in comparison to the standard kNN

Whilst unable to significantly rival the standard *k*NN results for correctness in large, high-class density data sets such as that of the Letter Recognition, the reduction in storage space when optimal hash numbers are chosen and the expandability of the memory provide advantage over usage of the standard LSH operation to determine the nearest neighbours for classification of unseen data points. In comparison with the Hierarchical Bloom Memory (Figure 52 and Figure 53), this variant may be slightly less time-efficient yet the storage space is significantly reduced and performance increased when the optimum number of hash functions (for compression i.e. less than the dimensionality) is employed.

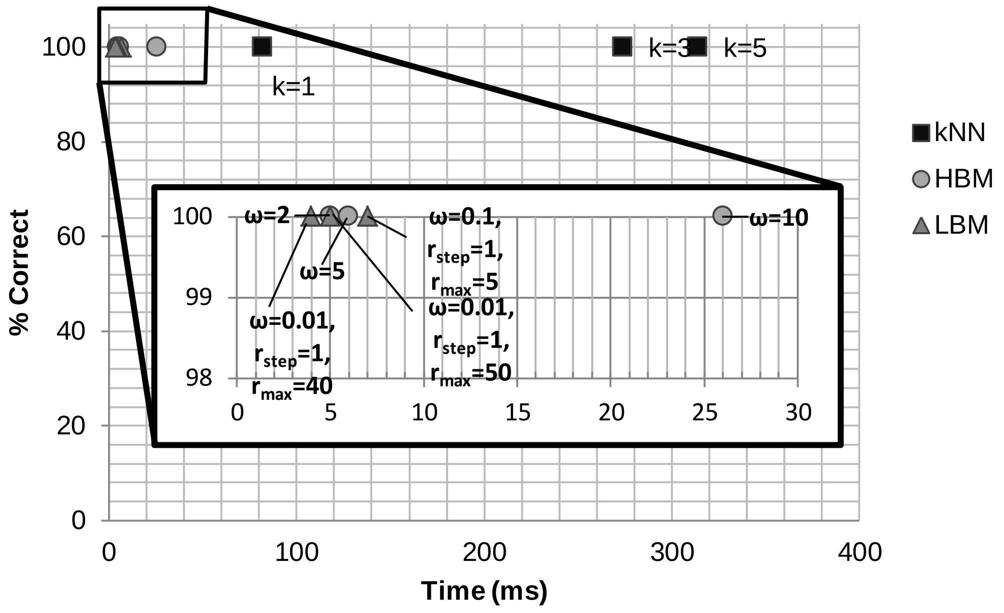


Figure 53 Fisher's Iris Data Set results with both Hierarchical Bloom Memory and Localised Bloom Memory variants in comparison to the standard kNN

6.4 Reasoning

The idea of Bloom Memory takes inspiration from the natural processes observable in the brain during the stages of object and element recognition, with inferably similar properties to some beliefs about the various memory centres present in adults. What has been identified through study of the brain is that recognition occurs in multiple areas at the same time [158 , 159 , 160], depending on the subject and context of the element being recognised. [161 , 162] highlight the common belief that recognition involves the interaction of multiple functional components; whilst not discussing the evidence to support the subsequent notion that these functions exist in various brain areas, it is noted that cerebral injury can impair the ability to accurately recall information that would otherwise be easily retrieved had all brain areas remained “intact”. Along with work such as that by Marr [163] and Van Belle [164], there is strong evidence suggesting that familiar objects are not represented by single active properties stored in some recognition centre in the brain but that a set of properties are needed in order for successful identification and that recognition can also depend on the physical arrangement of these properties as much as the properties themselves. With the Bloom Memory, the entire storage bank is required to be present in order to accurately and effectively return results as to whether an instance is stored in the memory – if a portion of the bank is lost, the memory system is rendered ineffective in much the same way as trauma would affect natural recognition. The capability for deletion of an instance in the memory can be likened to selective memory or some form of approved forgetfulness, where it may be the case that the storage in the memory requires reassessing for optimisation and so unused or irrelevant material is disposed of to make room for more useful elements. Each hash function must also return the same value of membership in order to successfully acknowledge the presence of an instance in the memory, again mirroring the theory that in humans all evidence must be available in order to determine the identity of an element or object. With the pseudo-random outputs of the hash functions for the resolution-based memory there is also similarity to neuronal activation in the human brain, given that there is no consistent theory of why specific neurons in the brain activate when they do upon the observation of an instance and they are not always localised to one area in a brain region. Conversely, the localised hashing scheme of the latter memory also mimics activation of neurons in local regions when similar instances are observed – enforcing connections in that area.

The storage of decreased resolution instances which generalise multiple higher-resolution instances is given credence as an algorithmic physiological equivalent, as in the literature it has also been shown that considerable modification of an image to be recognised does not necessarily limit our ability to determine its identity. [165] and [166] worked with extremely low resolution and degraded images, yet reached the conclusion that even with the reduced visual information the recognition rate was still high, thus indicating that class identification is still possible after a significant reduction in data resolution. [167] noted that the ability to tolerate degradation in data increases with familiarity – that is to say that missing or altered values in a data object will have less impact on its probability of recognition if the object is in context.

Taking on board the above points regarding similarity to biological memory and the effectiveness of each of the bloom-variants described here, the benefits of employing either in a suitable application are numerous when considering the possible alternatives. Whilst the Hierarchical Bloom Memory is simplistic in its implementation, from the Iris Data's experimental results it is obviously considerably effective at generalising the properties of this nonlinear and relatively high dimension data set and therefore results in superior speed for the same performance over other comparable classification methods and certainly improves on the storage requirements and time complexity of the standard *k*-Nearest Neighbours approach to classification in the high dimension space. The Letter Recognition test data is not so suitably classified with the HBM variation, due to the fact that at the lowest resolution it may be that a value is not covered by the appropriate range and therefore results in a lesser-certain classification. Improvements to the resolution determining stage would possibly increase the reliability of classification in significantly larger data sets (over the 4 dimensions for the successfully classified Iris Data) however some testing is required to determine the best parameters for each data set encoded with the HBM. The Localised Bloom Memory returns a higher rate of recognition than the Hierarchical variant for previously unseen points with the Letter Recognition Data, with a further significant advantage that the radius considered for the density calculations can be increased if necessary to further incorporate a larger area in the data space. The main advantages of both scheme's usage in the classification application is the returning of a value which can inform of the reasoning for a classification rather than simply acknowledgement of membership, i.e. if a point bears similarity to surrounding points, the density value reflects this and the classification is assigned accordingly. Both variants perform suitably well with the smaller Iris Data set, though it can be argued that due to the slight time cost difference the Hierarchical variant is more suited to smaller sets and thus the Localised more apt at classifying the larger sets as in the case of the Letter Recognition data – however theory would indicate that both can prove adept for classification when provided with adequate parameters for the data set in question.

An application in which there is a requirement to classify data that is intrinsically familial will therefore benefit from utilising these density-based memories so that possible outliers can be identified and further investigated if the application requires it. Both memories have their merits with respect to the classification of data, however it is also in the reduction of storage space for large, high-dimensional data sets that they possess a significant benefit.

6.5 Summary

This chapter has introduced two new storage, recollection and classification schemes for data sets that either have high dimensionality, or a considerable amount of data requiring significant compression in order to be of use in a classification application. The identification of the requirement for the development of these schemes arose due to the necessity for optimising the classification operation detailed in the previous chapter, whereby large data sets of high dimensionality were required to be significantly processed in order to increase their usability in the Verity behaviour monitoring application.

Where previously the classification decision came from the proximity to an already encoded/trained data point in a transposed, lower dimension data space (thus introducing an element of uncertainty with regards to training-data-positioning) – the principles of the above developed memories allow the data to be encoded in a relatively “pure” form, thus maintaining their localisations within the data set and allowing successive data points to be classified according to direct proximity in the high dimension space, with much less processing than any other method. Both methods exhibit an increased usability in the event of storage of subsequent data points possibly arising from correct classification, where the previous dimension reduction technique would require complete re-evaluation of data-describing prototypes in order for the new point to be included. The Bloom Memories here are more suited to real-time operation in cases where a *k*NN approach would prove beneficial yet storage space is limited.

Evidence of both schemes’ effectiveness was provided through commonly used data sets as before, with the results of the Localised Bloom Memory proving statistically adept at classifying unseen data points in high dimension data sets based on a *k*NN approach, where the Hierarchical Bloom Filter improves on classification speed in lower dimension data sets where parameters for nearest-neighbour determining are more inferable from examination of the training data.

Chapter 7
Behavioural Change Detection and the Combined Behaviour Monitoring System

This chapter details the behavioural change detection scheme employed within the Verity system in order to identify and alert the user or carer to issues which may be evident through analysis of the wearer's behaviour patterns over time, or even be immediately visible after they exhibit a specific series or sequence of states. The principles take their inspirations directly from those implemented in current industrial and personal monitoring systems, and are programmed into the graphical interface further detailed here, used to simulate real usage of Verity in a typical scenario. The simulation is shown to be suitable enough such that the algorithms and methods detailed in the preceding chapters provide adequate and accurate bases for inclusion within the actual Verity application.

The models documented within this thesis were developed with the intention of applying them to the behaviour monitoring system of Verity. For the effective operation of Verity in identifying anomalous or erroneous results regarding a user's behaviour, there must be a method of locating those states and state sequences exhibited by the user which are deemed out of context given their typical usage scenarios.

7.1 Detection of Possible Behaviour Issues

The HMM is a probability model: providing most likely sequences and states at individual time steps. It is driven by probabilities determined by numerous means: be it through testing, with adaptive optimisation algorithms or the newly developed techniques contained herein which identify probabilities based on patterns and/or clustering. The probability returned for the activation of any state not only provides information about that which is most likely, but it can also be interpreted as a value which classifies its degree of anomalousness, where low probabilities denote deviations from the norm [104 , 168].

The following identified types of anomalies are purely technical identifications and not associated explicitly with a specific reasoning for their existence according to the users' health status. Any errors arising from a deviation from what is expected are based solely on probabilities as determined by the models employed within Verity and are not explicit indicators of illness or behavioural concerns but are instead to be treated as early indicators of possible problems that the user may exhibit over extended observation once an error scenario occurs. These 4 instances do not by any means constitute an exhaustive list of all detectable problems with a user, and instead are identified as those of key significance to the Verity system in its current state. We desire not to develop another interfacing model for the detection of problems and instead place specific emphasis on utilisation of the HMM and its constituent parts as outlined in this work due to their collaborative ability to produce probabilities interpretable as certainty values.

7.1.1 Type$_1$ Anomaly

This first form of anomaly can be explained in terms of $\gamma_t(i)$: the Forward-Backward probability of the user being in a certain state i at time t given the observations and model; and $\hat{\delta}_t(i)$: the Viterbi probability of the user exhibiting the observations in state i at time t, given previous states and observations:

$$e(t) = \left| \max_{1\le i\le N}\left[\gamma_t(i)\right] - \hat{\delta}_t\left(\arg\max_{1\le i\le N}\left[\gamma_t(i)\right]\right)\right| \qquad 1 \le i \le N \qquad (95)$$

Due to state probabilities decreasing as the Viterbi algorithm progresses (the probability is for the state terminating the sequence), $\hat{\delta}_t(i)$ is required to be rescaled thus:

$$\hat{\delta}_t(i) = \frac{\delta_t(i)}{\sum_{j=1}^{N}\delta_t(j)} \qquad \begin{array}{c} 1 \le i \le N \\ 1 \le t \le T \end{array} \qquad (96)$$

The error therefore is the difference between the probabilities of the winning state determined by the FB method and the probability of the corresponding state from the Viterbi method. As the value of $e(t)$ tends to 0, the model's sequence determining methods will be producing similar estimations of the user's state. It can therefore be said that the certainty of the observations being produced by that state is high, given that according to both observation sequence and state sequence transitions both methods produce the same result. In the instances where the values differ significantly i.e. tend closer to 1, the certainty of the state producing the observations at that time step is lessened due to the larger disparity between both methods.

This type of anomaly detection algorithm can inform whether the observation is unlikely to have occurred in that scenario, even if the Viterbi algorithm makes sense of it as a part of the state sequence. In such a case, there is significant possibility that the Hidden Markov Model parameters are insufficiently defined for explicit detection of the state in question, with observation or state transition probabilities incorrectly approximated for use within this variation of the model for this particular user.

This is closest to the Output Observer (OO) method of fault detection [169], comparing the real, instantaneous observation sequence probability as determined with the Forward-Backward calculation with the nominal state sequence based probabilities from the Viterbi algorithm. The residuals are therefore computed as the difference between the probabilities of the current state determined through both methods to give a certainty measure. The residuals should be low when fault free (i.e. greatly certain if both methods return the same probability), therefore enabling the detection of faults through a simple threshold test [124 , 170].

In the event that the type_1 anomaly is detected in the system, it signifies that the model being used to describe the user is incorrectly defined, with the parameters providing the probability values for each state possibly over or underestimated based on human knowledge or inferred from the training cases, depending on the probability determining model currently employed. In the case that the Hidden Markov Model is employing the Fuzzy Inference System as a means of identifying observation probabilities based on the human-defined, weighted fusion of the sensor readings, one solution to the error is to check with the user directly through vocal interaction (as explained in Chapter 3) as a means of recalibrating the membership functions or rules governing that observation or transition probability.

If the observation probabilities are at fault in the Dimension Reduction and Classification scheme, there is likelihood that the set of training examples used did not encompass a situation where the observations described the determined state adequately enough, therefore the prototypes incorrectly surmised the data: leaving the subsequent unseen observations producing the error with an extrapolation of a probability from existing data rather than an explicitly defined one. What would be required in this instance is a re-training of the prototypes with this new value after similar confirmation with a user that they are not experiencing any other problems at that instant, perhaps due to the fact that as part of a state sequence the observed sensor values don’t match those expected and thus indicate an erroneous behaviour from the user.

Again in the Bloom Memory implementation of probability determining, the density of training points within the Memory Layer may not be significant enough to indicate a satisfactory probability of existence for the observation vector, therefore there is yet another requirement to insert this value into the memory after another confirmation process such that subsequent instances of this vector's appearance are accounted for.

7.1.2 Type$_2$ Anomaly

A type$_2$ anomaly is based further on the certainty of the winning state. If the probability of the winning state occurring (P_{win}) is close to the other states' probabilities, its certainty (ρ) is lessened due to the possibility that the observation falls close to a boundary used in the probability determining process: i.e. it has very little dominating likelihood of occurring in the current winning state if only a slight change in one of the sensor readings will result in the returning of another state. For all active states (with the exception of the scenario where the winning state is a single active state), the proximity to the mean of the probability over all those states is calculated. When the probability is close to the mean, the instance can be deemed uncertain (ρ tends to 0).

$$\begin{gathered}\mu = \overline{P\left(q_t = S_i\right)} \\ \left\{i : P\left(q_t = S_i\right) > 0\right\}\end{gathered} \qquad 1 \le i \le N \tag{97}$$

$$\rho = P_{win} - \mu \tag{98}$$

In the case where the value of ρ falls within a specified threshold to indicate significant uncertainty it can be said that state transition and observation probabilities are again misdefined in the behavior recognition model. Unlike the previous error however, this would not arise due to an observable anomaly with the user (such as exhibiting a behavior typical of another observation) as the parameters involved are all single model-based, i.e. they don't compare different state determining methods and therefore are a result of the defined probabilities being imprecise. Here the transition probabilities can be reassessed to ensure the weighting of an observation after being seen in a certain state is significant enough to overcome the threshold value and enhance certainty, or again the observation probabilities can be checked and the appropriate probability determining scheme updated accordingly as indicated for the previous anomaly.

7.1.3 Type$_3$ Anomaly

An equally likely scenario develops when the observation witnessed does not belong at all in the sequence – that is, it is not emitted from a state which can be transitioned to from the current state, which, in our ergodic, interconnected-state model would indicate the presence of a new unseen state or observation from a currently defined state not seen or considered before. This will result in the model returning an error given that the most likely state now has a probability of 0 due to the insufficiently modelled observations, behaviours and their transitions. Detecting such an error primarily requires monitoring of the relevant observation probability determining scheme returning the values to the Hidden Markov Model, where if the probabilities over all states having seen observation O is 0, the inference is that the model has not seen such an observation before and therefore requires either reassessing or triggering an alert as per the standard procedure of user interaction (Verity communicating via speech), with alert type$_3$ as the cause:

$$P_{all}\left(O_t\right)=\sum_{j=1}^{N}b_j\left(O_t\right) \qquad 1\leq t\leq T \qquad (99)$$

An instance where this form of anomaly could arise is likely if not all possible observations and associated states were captured during the training phase, or if the user exhibits a behaviour typical of an unprogrammed state which is subsequently required to be included. In an instance where the observation is indicative of a serious issue with the user, i.e. a stroke or heart attack indicated by increase in temperatures and heart rates, the observation would either trigger this type of anomaly due to the state not having been seen during training, or the type$_4$ anomaly described below which detects a difference in detected state at an instant and the detected state as part of the entire sequence.

For the scenario where type$_3$ anomalies are detected, the model requires updating to include the new, “safe” state if the user agrees that it is a standard state in which they can be observed (or parameters of current states adjusted to incorporate the new observation as explained for the previous anomalies), or the alerting of an external carer or service to the fact that the user is exhibiting erroneous behaviour. Both solutions are executed through communication with Verity through the speech recognition module.

7.1.4 Type$_4$ Anomaly

A type$_4$ anomaly is a slight variant on both type$_3$ and type$_1$ anomalies and can occur simply when the state at a time step differs for each state determining method within the HMM i.e. the Viterbi state (q_t*) and the winning state according to pure observation probability ($b_j\left(O_t\right)$) do not match: $q_t* \neq \arg\max_{1\leq j\leq N}\left[b_j\left(O_t\right)\right]$. Factoring in the probability based purely on the observation ($b_j\left(O_t\right)$) affords the ability to assess whether or not the observation and determined state are equivalent in their likelihoods, as suggested above: if the observation probability is highest for perhaps the state of *Running*, yet the determined state according to the Viterbi method (q_t*) returns *Sleeping* with higher probability over its *Running* probability, this may in fact indicate a period of distress for the user such as in the instance of a heart attack or some other such observable problem. The probability from Viterbi is rescaled as in (96) to give $\hat{\delta}_t\left(j\right)$, which is then compared with $b_j\left(O_t\right)$ rather than the Forward-Backwards' $\gamma_t\left(j\right)$ as in type$_1$, after the condition of their output state-difference is met:

$$C = \begin{cases} \left|\hat{\delta}_t\left(j\right) - b_j\left(O_t\right)\right|, & q_t* \neq \arg\max_{1\leq j\leq N}\left[b_j\left(O_t\right)\right] \\ 0, & \text{otherwise} \end{cases} \qquad 1 \leq t \leq T \qquad (100)$$

$$e\left(O_t\right) = \begin{cases} 1, & C > k \\ 0, & \text{otherwise} \end{cases} \qquad (101)$$

Where here k is some decimal constant in the range $\left[0,1\right]$ used as a threshold to identify whether or not the difference between the two differing states is significant enough to trigger an alarm. The returned states must first be identified as different before the identification of by how much, however, so that instances where the same state is determined but with differing probabilities are not misclassified as errors when in fact they are just “unlikely”. It is these misclassifications that are handled by type$_1$ and type$_2$ anomaly conditions, where the likelihood of the state is addressed rather than the disparity between two differing states.

As well as identifying possible occurrences of serious health problems such as heart attacks, when viewing the entire state determining process as a whole sequence - perhaps after a significant period of monitoring - this type$_4$ anomaly will prove quite useful for the detection of behaviour changes as it has the potential to highlight instances where the user exhibits behaviour not considered likely according to the transition probabilities programmed at the start of the process. Where a non-threatening state is observed (i.e. the user has in fact begun a higher level of exertion immediately from a rest period, thereby triggering a *Sleeping* to a *Running* state change) then the transition probability between the two requires amending to allow for such an observation sequence, through the interactive speech confirmation process implemented within Verity.

7.1.5 Further Anomaly Detection

One suggested mode of operation for behaviour classification and associated error detection would be to break the monitoring process down into “blocks”, with a single block representing a certain time period in the user’s day during which observations and states can be taken and compared with those from previous days, where all states and the above error types calculated for a block on one day are simply compared with one another for discrepancies – the theory being that over a period of time if all errors are seen within similar periods on different days, the user is exhibiting a constant, unusual behaviour that needs further investigation or reconsideration of the model and parameters used.

Thresholds can be set which trigger alarms when the comparison between states, their probabilities or error values in a specified time period or “block” is too large to be acceptable as part of a standard routine. A block, for example, may be considered to be variable and consist of an expected period of activity or linguistic description of the part of the day to which it is associated. Figure 54 describes a possible scenario for a user that forms a full description of their day.

The blocks describe the periods of the day tailored to the user, thereby providing the system with a standard expected series of observations based on the time period with which to compare previous results and those within the block itself. The sleep period, for example, would comprise primarily of low acceleration, low heart rate and a relatively constant temperature – properties assumed from experience and knowledge of the scenario. However, the Morning, Afternoon and Evening periods are significantly variable on a daily basis and may have to encompass activities such as shopping, cleaning or situations like bathing and washing up. These variable periods would cause the most conflict on a daily basis if compared with previous days, as with any user there is no guarantee of a “set” pattern of activities.

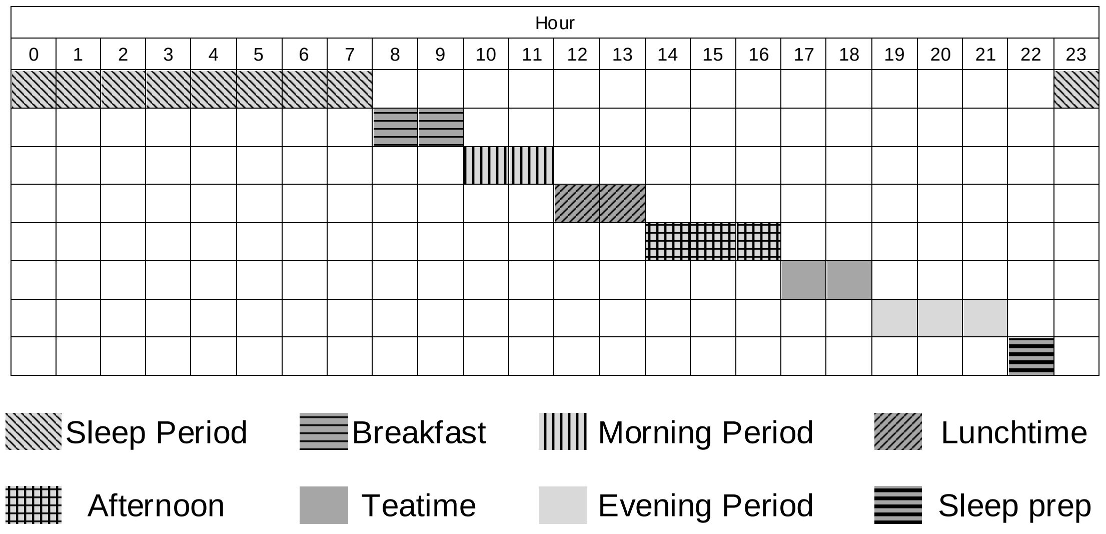


Figure 54 An example breakdown of a user's day into descriptive blocks

These block descriptions have the ability also to scale to larger time frames where for analysis such periods as weeks, months, seasons etc. can be assessed with previously captured periods for all the error types described above and possible issues identified with the user as a result (health issues or care requirements, for example). Whilst some detectable anomalies such as types 1 and 2 can be considered to be errors with model parameters more so than with the user, the type 3 and 4 anomalies would be more serious if detected immediately as they indicate a problem scenario is currently unfolding that requires immediate attention. However, when the probabilities as determined by the type$_4$ anomaly are compared over time – perhaps with this block method – there may be seen to be a slow change in the user’s behaviour as they move more to exhibit one state at a specific time instead of another, thereby indicating a general shift in behaviour patterns that either need investigation as an indicator of slower-onset health issues, or simple model adjustment with supervision from a knowledgeable professional who can make sense of the cause of change. As discussed in Chapter 2, time-series and sequence analysis of a user's daily (or weekly, or other large time frame) behaviour in order to determine errors and possible faults is a large field of research that requires more in-depth analysis than is addressed in this work, however with a starting position such as this identified with the above errors and reasoning, there is a significant enough basis on which to develop further techniques which utilise our developed models to build a complete behaviour monitoring system for the elderly and those requiring constant care.

7.2 The Combined System in Simulation

Throughout the research work detailed in this thesis the primary aim has been to develop the decision making and probability determining mechanisms for inclusion in a personal and wearable device, Verity, used for the constant behaviour and state monitoring of an elderly or infirm user. As a result, the techniques developed have been tailored specifically to the problems identified at each stage of the practical system's development and those which have arisen as the result of the implementation of a previous technique.

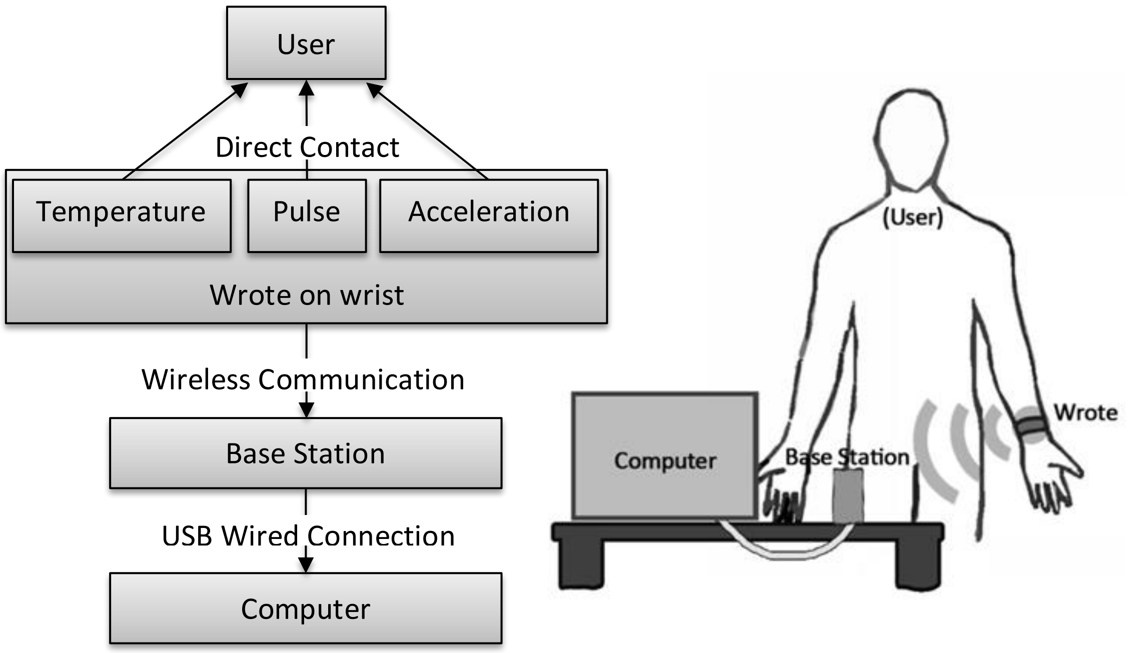


Figure 55 Combined Behaviour Monitoring System test configuration

For ease of algorithm testing, visualisation and debugging, the multiple algorithmic systems developed for the research were combined in software for simulation to assess their inter-operability such that once the device is complete, they can simply be programmed into their respective device locations ready for use with the observation-gathering sensors.

A combination of software environments were used in order to optimise the simulation process, with different programmes employed according to their suitability for the task at hand. MathWorks' Matlab is most suited to the rapid visualisation of data in graphical format, with the object-oriented language of C# programmed with Microsoft's Visual Studio allowing for easily modifiable and application specific graphical interfaces for complex algorithms running on a standard home computer.

With the Wrote situated on the user's wrist transmitting sensor readings wirelessly through the paired-Sensium CC981 control chips to the Verity base station – which is connected to the computer via a COM port through USB (Figure 55) – the Visual Studio software began to update the specially designed graphical interface with which the observations and state results can be viewed in real-time.

With the system now ready for operation, the readings from the sensors were gathered and assessed in order to formulate a series of initial rules and to define the parameters for transition, as per human knowledge. The data gathering with Verity resulted in a series of 30 data readings (Table X) obtained from the sensors over the course of operation, each with an attributed state which was observed to have produced such readings. Based on these readings and the known state that was producing them, the membership functions and rules were constructed using human knowledge of the situation/observation relationship.

It is worth noting that the readings returned by the preliminary test with Verity do not match those used during the simulated experiments in previous sections, with some ranges and values exhibiting considerable differences to those originally used. In Table X it can be seen that the ambient temperature does not fluctuate at all, primarily due to the fact that all states and behaviours were observed in the same environment. Contact temperature does not centre on an expected 37°C, but instead is around 10°C lower due to the thermal characteristics of the sensor and its placement on the user's wrist. In practice, this does not affect the system as all values can be seen as arbitrary and only used as indicators of a state, with scale being inconsequential. The pulse also varies by a minimal amount, though a marked increase can be seen in the higher-motion instances of the later states; the motion here is also greater than the original range specified and serves as a significant indicator of state – with the earlier, more docile-appearing behaviours being reflected with a lower value.

Table X Verity Data obtained during Testing and Simulation

No.	***Ambient***	***Contact***	***Pulse***	***Motion***	***Orientation***	***State***
1	28.699	28.776	76.142	0.000	1	2
2	28.699	28.776	76.142	0.000	0	1
3	28.699	28.818	80.213	0.000	0	1
4	28.699	28.818	80.213	0.000	0	1
5	28.699	28.818	80.213	0.000	0	1
6	28.699	28.838	81.967	0.256	0	3
7	28.699	28.838	80.213	0.170	0	1
8	28.699	28.838	81.967	0.114	1	2
9	28.699	28.838	81.967	0.114	1	2
10	28.699	28.849	81.967	0.115	1	2
11	28.699	28.849	81.967	0.172	1	2
12	28.699	28.849	81.967	0.172	8	2
13	28.699	28.838	81.967	0.598	8	4
14	28.699	28.838	81.967	1.084	8	4
15	28.699	28.849	81.967	1.170	8	4
16	28.699	28.849	81.967	0.827	8	4
17	28.699	28.849	81.967	0.458	8	3
18	28.699	28.828	81.967	0.458	8	3
19	28.699	28.828	81.967	0.458	8	3
20	28.699	28.797	81.967	0.458	8	3
21	28.699	28.797	81.967	0.515	8	3
22	28.699	28.797	81.967	0.516	8	3
23	28.699	28.683	81.967	0.686	8	4
24	28.699	28.683	81.967	0.686	10	4
25	28.699	28.683	81.967	1.627	10	4
26	28.699	28.662	81.967	1.799	10	4
27	28.699	28.704	81.967	2.370	0	4
28	28.699	28.704	81.967	2.828	0	4
29	28.699	28.704	81.967	2.484	0	4
30	28.699	28.704	81.967	2.484	0	4

The orientation value (Figure 56) was here included as an indicator primarily for debugging purposes and would only serve in the real system as a marker for a healthcare professional to ascertain changes in the user's orientation during each behaviour, with the later possibility of including some conditions which assist in the detection of a *Sleeping* or “collapsed” user perhaps based on the position of their wrist after a period of inactivity. To exclude the orientation value from the fuzzy rules in this first instance was a decision based on the fact that a wrist may be in any location on any axis during any observed behaviour, therefore to explicitly model with rules would be unwise and may put too much bearing on a behaviour according solely to the position of the Wrote.

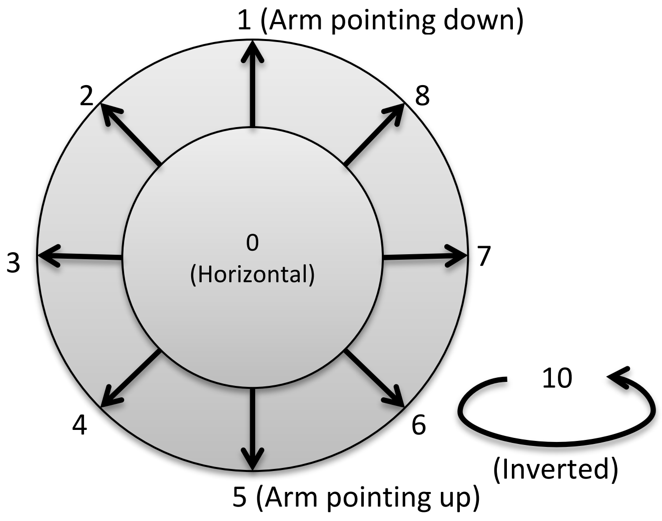


Figure 56 Orientation of the device and corresponding value

The transition parameters of the Hidden Markov Model remain the same, with a_{ij} specified as in (102) and given the observed starting state, the initial state probability vector π is as in (103) where there is an observed higher likelihood that the starting state is *Standing* over all others.

$$a_{ij} = \begin{bmatrix} 0.45 & 0.35 & 0.20 & 0 & 0 \\ 0.25 & 0.35 & 0.30 & 0.10 & 0 \\ 0 & 0.35 & 0.20 & 0.35 & 0.10 \\ 0 & 0.10 & 0.25 & 0.40 & 0.25 \\ 0 & 0.10 & 0.15 & 0.25 & 0.50 \end{bmatrix} \tag{102}$$

$$\pi = \begin{bmatrix} 0.1 & 0.2 & 0.4 & 0.2 & 0.1 \end{bmatrix} \tag{103}$$

Using the Verity-gathered sensor results, the membership functions used in the original experiments were amended to arrive at those shown in Figure 57, which are used in the governing fuzzy rules (again specified here, with reasoning as in Chapter 4):

0. **Sleeping**: Ambient Temperature is *Hot*, Contact Temperature is *not Cold*, Pulse Reading is *Normal* and Acceleration is *Nil*
1. **Sitting**: Ambient Temperature is *Normal*, Contact Temperature is *Normal*, Pulse Reading is *Normal* and Acceleration is *Nil*
2. **Standing**: Ambient Temperature is *not Hot*, Contact Temperature is *Normal*, Pulse Reading is *Normal* and Acceleration is *Nil*
3. **Walking**: Ambient Temperature is *not Hot*, Contact Temperature is *Normal*, Pulse Reading is *not Low* and Acceleration is *Minimal*
4. **Running**: Ambient Temperature is *not Hot*, Contact Temperature is *not Cold*, Pulse Reading is *High* and Acceleration is *High*

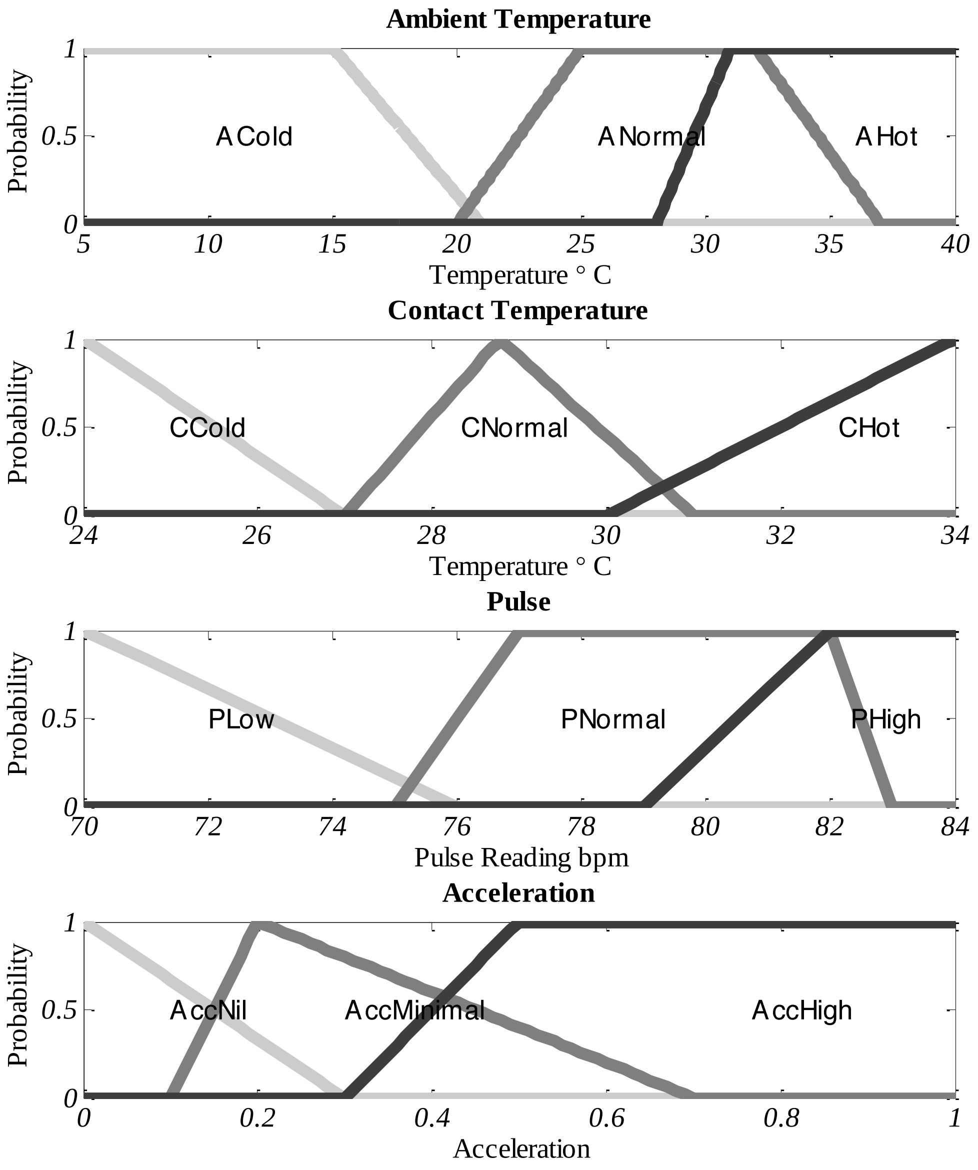
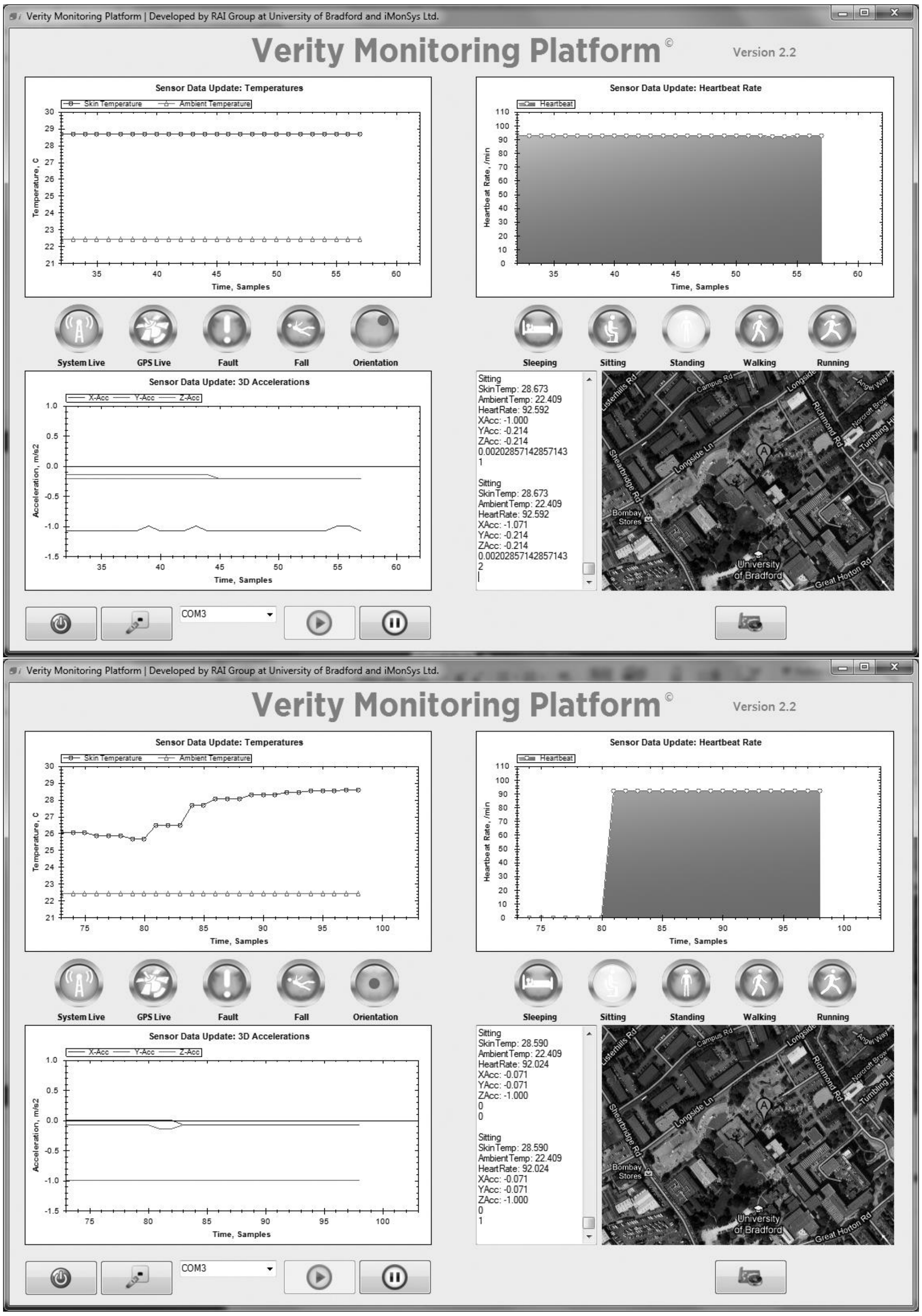


Figure 57 Membership functions for fuzzy inference system used in simulation experiments



```
Sitting
SkinTemp: 28.673
AmbientTemp: 22.409
HeartRate: 92.592
XAcc: -1.000
YAcc: -0.214
ZAcc: -0.214
0.00202857142857143
1

Sitting
SkinTemp: 28.673
AmbientTemp: 22.409
HeartRate: 92.592
XAcc: -1.071
YAcc: -0.214
ZAcc: -0.214
0.00202857142857143
2
```

```
Sitting
SkinTemp: 28.590
AmbientTemp: 22.409
HeartRate: 92.024
XAcc: -0.071
YAcc: -0.071
ZAcc: -1.000
0
0

Sitting
SkinTemp: 28.590
AmbientTemp: 22.409
HeartRate: 92.024
XAcc: -0.071
YAcc: -0.071
ZAcc: -1.000
0
1
```

Figure 58 The Verity Monitoring Platform Graphical User Interface (GUI) for testing and debugging. Top image indicates a Standing state, with the lower image displaying a Sitting state

Figure 58 shows the graphical interface during operation as viewed from a debugging perspective. In standard operation, the user would not see such a screen and instead be effectively "blind" to their data and the observation-dependent states: only being made aware of issues and/or states if requested or if the system requires some input by them for qualification of a result.

The figure shows the sensor readings from the Wrote with only a short lag from the data retrieval stage to the processing of the results. The key indicators are the Heartbeat, Temperatures and Acceleration which are located anticlockwise from top right. The GPS location is envisioned to be situated on the same screen (here shown in the bottom right) so that the person conducting the monitoring is able to view the whereabouts of the user and perhaps form a better idea of why at certain stages the readings present as they do i.e. an incline may indicate why the user is presenting a high temperature and fast heartbeat, whereas a city-based, indoor shopping location may explain high acceleration and/or temperature.

Using the newly defined rules and memberships, the same sensor readings are submitted as observations to Verity's state determining scheme using the FIS, which returns the states being exhibited by the user wearing the device. In the first example of the system returning state values, (the upper image of Figure 58) the orientation of the device is shown to be within the top right section of the indicator and therefore signifies to us that the user is upright, with their arm by their side and pointing towards the ground (a value of 2, 1 or 8 indicates such a position, as shown in Figure 56). Along with the other observations of a low acceleration and relatively steady heartbeat and temperature, the conclusion reached is that the user is *Standing* given the observations in isolation (using both the Forward-Backward procedure of state determining and the raw FIS winning state) and *Sitting* based on the results obtained from the Viterbi procedure – taking into account previous states and transitions. Using our defined error detection scheme this disparity flags up as $type_1$, $type_2$ and $type_4$ anomalies given the mismatch of states and the closeness of each of the respective determined possible states' probabilities. In the case of the $type_1$ anomaly the disparity between probabilities for *Standing* and *Sitting* is low for both state determining methods, yet significant enough to trigger the detection with a threshold of 0.2, where the $type_2$ anomaly has a value close to 0 - also indicating close probabilities for the two states and therefore a lesser certainty that it is a winning state based on the Forward-Backward method.

The type_4 anomaly triggers with the initial condition of a differing state from both the FIS and the Viterbi method: with the Viterbi accounting for transitions, the FIS outputs a value of *Standing* based on the observation and therefore the error flag $e(O_t)$ is activated when the applied threshold of 0.2 is again reached. During the testing procedure this input observation was the only such error to occur due to the other observations being typical for their states, and no thresholds were otherwise exceeded. The GUI indicates the results of the different methods also, as can be seen with the lit icons on the right of the screen and in the debug window in the lower centre, where the values for each sensor are also provided to a higher accuracy for the use of a professional or for debugging purposes.

As the user's position changes from *Standing* to *Sitting*, the GUI reflects the fact (the lower image of the figure) by visualising the change from an upright orientation of the device to a horizontal orientation, thus indicating that the user now has the device parallel to the ground as is indicative of a user with their arm perhaps resting on a chair. This change, coupled with the observable change in temperature, motion (acceleration) and heartbeat has resulted in the conclusion with both state-determining methods that the user was *Sitting* at that instant. It is likely that the sequence-state of *Sitting* now observed is now more certain due to the instantaneous result from the FIS being the same as both Viterbi and FB methods, and subsequent states will require feasible transitions from *Sitting* before any decision is made as to the user's behavioural state with the Viterbi method. The gathered readings and associated states were then submitted to the State Visualiser which executes (offline) the Dimension Reduction and Classification procedures detailed in Chapter 5. Figure 59 shows the visualiser window that has successfully taken the readings from their initial 5 dimensions to the more easily viewable 2, without loss of structure and resulting in the creation of 4 linearly separable state clusters with which subsequent classification of unseen data points can occur (note that the state of *Sleeping* was not observed in this test of Verity and data gathering procedure due to the conditions indicating such a state not being easily obtainable during testing).

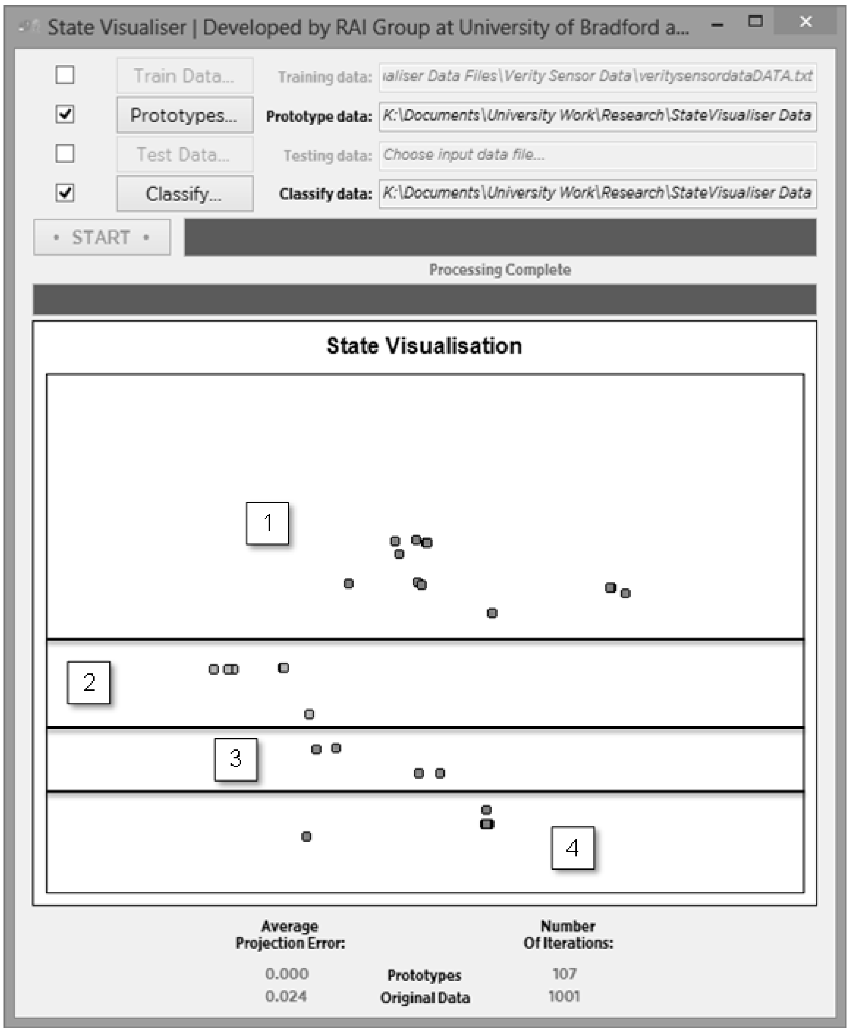

State Visualiser | Developed by RAI Group at University of Bradford a...
Train Data...
Training data: ıaliser Data Files\Verity Sensor Data\veritysensordataDATA.txt
Prototypes...
Prototype data: K:\Documents\University Work\Research\StateVisualiser Data
Test Data...
Testing data: Choose input data file...
Classify...
Classify data: K:\Documents\University Work\Research\StateVisualiser Data
• START •
Processing Complete
State Visualisation
1
2
3
4
Average Projection Error:
Number Of Iterations:
0.000
Prototypes
107
0.024
Original Data
1001

Figure 59 State Visualisation with results obtained during real-time Verity operation. States are separable with the lines on the figure and are labelled accordingly.

With the data set now having been processed and the prototypes for the state determining identified, the perceptrons which generalise the states are created and trained in order to allow for distinction between, and classification of, subsequent unseen data points. Table XI shows the weights produced using the Verity data from the live testing, with Table XII showing the results of classification with the perceptrons for previously unseen data points. The combined HMM and FIS is now updated in order to incorporate the new neural network: the dimension reduction scheme identifies a correlation between data and state in the lower dimension once sufficient data has been obtained with which to train, resulting in the removal of the original FIS used to classify the unseen data. Now, the data would be presented to the perceptron network after dimension reduction and a probability of occurrence given with which to use in the HMM rather than the previous Fuzzy probability value.

Table XI Weights from training perceptrons with Verity Data

State Perceptron	***Weight 1***	***Weight 2***	***Weight 3***
1	-3.3668	-3.7487	-5.9766
2	-6.5034	-6.1405	-37.5824
3	-11.3990	-12.2098	-0.4053
4	16.9053	18.2461	-4.3193

Table XII Result of using trained weights on unseen data points

Ambient	***Contact***	***Pulse***	***Motion***	***Orient.***	***Actual***	***Result (Probability)***
28.699	28.838	80.213	0.000	0	1	1 (0.98)
28.699	28.838	76.142	0.170	0	1	1 (0.98)
28.699	28.849	81.967	0.114	8	2	2 (0.99)
28.699	28.797	81.967	0.458	6	3	3 (0.98)
28.699	28.662	80.213	1.799	0	4	4 (0.98)
28.699	28.704	81.967	1.799	10	4	4 (0.99)

Table XIII Result of classification with Localised Bloom Memory for unseen data points

					State Probability				
Ambient	***Contact***	***Pulse***	***Motion***	***Orient.***	***1***	***2***	***3***	***4***	***Actual***
28.699	28.838	80.213	0.000	0	0.985	0.011	0.003	0.001	1
28.699	28.838	76.142	0.170	0	0.501	0.369	0.000	0.129	1
28.699	28.849	81.967	0.114	8	0.000	0.782	0.198	0.020	2
28.699	28.797	81.967	0.458	6	0.005	0.385	0.498	0.112	3
28.699	28.662	80.213	1.799	0	0.385	0.188	0.035	0.392	4
28.699	28.704	81.967	1.799	10	0.000	0.003	0.014	0.983	4

The same training instances were submitted to both Bloom Memory schemes also, in order to ascertain suitability for inclusion within Verity. Table XIII shows the results of the more successful Localised Bloom Memory implementation, returning 100% classification correctness on the same unseen data as used in the previous experiment.

Table XIV details a comparison between the 4 different state probability determining methods, with key parameters that resulted in the best classification rates during experimentation. With the FIS the training time is indeterminable due to the nature of the system: knowing the scenario and viewing the initial data the rules are constructed linguistically and programmed accordingly, with amendments made to ensure that the membership functions most adequately surmise the situation. Classification takes little time, as all that is required is determining which sensor membership range a reading falls into, with the basic Mamdani minimum decision made as to which state most likely produces the readings. States are identified correctly primarily because the membership functions and rules directly describe the system, therefore errors are evident only with unseen observations and incorrectly modelled behaviours.

Table XIV Best recognition rates and performance times for the 3 state determining methods classifying the previously unseen test data in isolation

Method	*Key Parameters*	*Training Time (ms)*	*Classification Time (ms)*	*Correct (%)*
Fuzzy Inference System	5 rules, 3 membership functions per sensor	-	17	100
Dimension Reduction with Linear Perceptrons	$tolerable\_loss = 0.1$, $alpha_{\min} = 0.02$, $alpha_{\max} = 0.5$	5713	154	100
Hierarchical Bloom Memory	$b = -1$, $p = 2$, $\omega = 5$, $h = 10$	11	2	67
Localised Bloom Memory	$r_{step} = 1$, $r_{\max} = 10000$, $\omega = 0.001$, $h = 30$	32	94	100

Despite being the longest in its classification process, the classification with dimension reduction scheme also took under 1 second, however it is in the training (projection) of the prototypes that we cement this as the most time consuming method over all of those developed with an outlay of nearly 6 seconds to prototype and project the 30-member training set. Classification is again 100% accurate for the experiment, with the returned membership values tending very close to 1 due to the certainty through dimension reduction that the unseen data points fall within the newly created linear boundaries between classes.

Submitting the same data to be trained with the Hierarchical Bloom Memory takes just 11ms with the identified optimum parameters, returning the lacklustre 67% correctness for classification in only 2ms. The returned incorrectly classified states may be deemed a worthwhile loss in accuracy based on the speed of classification if the scheme in which the HBM was employed merely "estimated" a state membership, however for the Verity application it may be seen as too inconclusive for use in behaviour monitoring.

The Localised Bloom Memory perhaps provides the best result over all methods, with a short training period and acceptable classification speed, the 100% correctness and format of probability values seems most appropriate for use in the proceeding Hidden Markov Model. The hash number used to produce the useful result was 30, a number significantly higher than the data's original dimensionality, producing a number of memory layers with more space requirements than the raw data – though the higher performance in classification over the other methods comparatively (accounting for both training time and correctness) sets the LBM as the most suitable candidate scheme for future inclusion within the Verity system.

7.3 Summary

This chapter has detailed the final elements and aspects developed for inclusion within Verity, and the combination of the previous chapters' methodologies for enhancing and achieving the goals intended by the behaviour monitoring device have gone some way to explaining and exhibiting the effectiveness of the device in simulated operation. Four variations of detectable anomalies and errors possible with the Hidden Markov Model implementation in a behaviour monitoring application were explicitly identified and possible causes suggested, with scope for the detection of other erroneous scenarios discussed and action procedures detailed for the event of such situations arising.

All methods developed within the research for the behaviour monitoring project were tested as part of simulated experiments with real data gathered from the Verity device, with each method exhibiting strengths and weaknesses in different areas of operation. The Fuzzy Inference System was identified as the most reliable scheme for operation in the first instance where states are determinable by human analysis of observation data – perhaps as a means of providing the initial Verity system with a working model on which rules can be expanded and tailored to a user before a more statistical and methodical approach is employed for the determining of probabilities in a lower dimension.

The dimension reduction scheme not only proved effective in the visualisation of the high dimension behaviour data – giving visual representation of observations that fall outside of the majority of state data – but in the speed of classification once the training data is learnt and projected. The combination of these two points further the method's practicality in the behaviour monitoring application when a secondary user requires access to the gathered data for diagnostic purposes, however in comparison to the latter two schemes developed perhaps doesn't exhibit a greater efficiency in the classification process.

The two Bloom Memories require no model inference from the data and can therefore be populated almost instantaneously in comparison to the other methods, returning better and more practical probability results than the previous (though not necessarily more applicable values over the initial FIS based on the human knowledge). However the Localised Bloom Memory is speedy, efficient and easily usable: proving most adept at classification of the sample Verity data thus serving well as a starting scheme on which to base the behaviour monitoring system.

Chapter 8
Conclusions, Observations and Further Work

In this chapter conclusions are drawn from the research conducted for the project and documented in the previous chapters, discoveries and developments are reflected upon and interesting elements which arose during the process are identified with suggestions provided such that further work may possibly be undertaken based upon the findings.

For the research were developed three different probability determining methods for inclusion within the Hidden Markov Model as a means of replacing standard observation probability definitions, with each scheme also proving suitable for applications other than behaviour monitoring where the updating of probabilities during use is required. Here the most significant contribution to the behaviour monitoring and state identification and classification fields is identified to be the Localised Bloom Memory, with the other developed methods' benefits and drawbacks highlighted as properties suiting them to various aspects of the monitoring process. Further work is suggested as a means of continuing the research based on the findings detailed in the previous chapters.

8.1 Conclusions

It is worth noting that the majority of the techniques and methods developed for this research and documented in this thesis were programmed and tested in isolation from the intended hardware device, in a simulated environment. The key operational parameters and probabilistic Hidden Markov Model at the heart of the Verity system had been identified in the preliminary stages of the research process, with industrial partner iMonSys stipulating that the result of the research should be a pervasive device capable of providing a monitoring platform for the elderly and infirm. Incorporating the methods researched here into the device requires further consideration insofar as adapting the language in which they are written in the test software to that of the language of the hardware, and will foreseeably operate effectively in symbiosis with the on-board sensors for which they were purposefully designed.

In the creation of the combined Hidden Markov Model and Fuzzy Inference System we have integrated two probabilistic methods to form a single, usable model for the first instance of our behaviour monitoring application. The identification of the HMM as ideal for the task of sequential behaviour determining proposed a problem for the incorporation of the multiple sensors which were to provide observation values as a means of evidencing a state's occurrence, as no such models existed in the literature to deal with the issue. Fuzzy Inference Systems by their nature incorporate multiple sources of information which combine to produce a single, useable value which in our case resulted in the observation probability previously described by an explicitly defined matrix in a standard HMM. The outcome is a system transferable to many applications requiring the fusion of multiple input observations into a single value to infer a state belief as part of a larger sequence of events or instances. Comparably, using a continuous distribution HMM to model the observation values obtained with the sensors results in ineffective combination of evidence available to the system when again further consideration is required to determine how much weighting each sensor requires towards the final probability value; the behaviour monitoring application here benefits due to the element of human reasoning forming the basis for the fusion step and informing of the independent weightings of the sensors due to their contribution to the fuzzy rules.

The 4 schemes created to detect possible instances of a behavioural change with an elderly user take the HMM and its state sequence determining methods as a means of identifying issues, viewing the probabilities of state occurrences as indicators of certainty when compared with other state probabilities generated by the model. Each error type is also transferrable to other implementations of the HMM in different applications because of their definitions, i.e. with state certainty as calculated in the type $_2$ identification, any state determined by an HMM can be evaluated for reliability against other possible states and thus trigger further investigation if required. The inclusion of the 4-type error detection scheme for Verity provides a complete framework on which to build successive detection methods, allowing for an initial version of Verity to exist in order to identify other behavioural change indicators detectable with the developed probabilistic model.

It can be concluded from the development of the Curvilinear Distance Analysis for Linear Classification technique that utilisation of a manifold’s global distance is most appropriate when considering the topology and interlinking of state clusters, especially when maximum separation is required for the purposes of training with a linear classifier. Analysis of comparable dimension reduction techniques in the literature review highlighted that previous developments suffered from drawbacks in their projection and successive interpolation steps; although not perfect the CDALC approach to the issue provides a guaranteed means of separating data clusters inherent to the nonlinear manifold and ensures that no “parasitic links” which cause stretching and tearing of a data set occur, certainly not in the sets of dimensionality of up to 13 tested in this research. Training a linear classifier such as the perceptron used here is much easier on sets with a visible and determinable plane of separation, as they allow for optimum placement of the line function for classification within an uninhabited data space where a classifier operating on nonlinear data must find a position minimising an error value which may never reach 0. This is evident through the testing with the multi-layered neural network which failed to classify each set of the test data to an acceptable degree of accuracy for our system, with the dimensionally reduced set instead succeeding in all trials and thus proving the CDALC is effective for binary classifiers.

The dimension reduction scheme does require storage of the raw high dimension data alongside the lower dimension, as with an expanding system the reduced dimension data needs to be retrained to be inclusive of points seen since the last training period. This poses issues for devices like Verity with low storage memory ability desiring to operate entirely online, so identification of the Bloom Filter and the Hierarchical and Localised variants of our developed memory utilising the filter principles affords a more optimal classification scheme which can operate directly in the high dimension. Comparing the approach to the standard *k*-Nearest Neighbours as a benchmark does show that the memory variants classify the lower dimension data sets in a much faster time (less than 6% in one instance). However, a slight sacrifice must be made for data sets of a larger dimensionality which do not garner as high a result in the classification, but do remain capable of being processed at a high speed for “approximate” membership queries. It is these instances which would then require further investigation through methods not detailed or explored within this research. When applied to the Verity system and the classification values used in the HMM as probabilities of membership to a state, the Localised Bloom Memory performs most accurately over the entire collection of probability determining methods and succeeds to be the most suitable to be incorporated into the first Verity implementation.

8.2 Observations

The behaviour monitoring field – especially when considering devices aimed at the elderly – is one that is ever-expanding as a result of constantly emerging new technologies and the further identification of particular subsections of the elderly population who require differing standards of care.

The probability determining and classification schemes developed here were born of identifying a need to deal with the nonlinear and high dimension data commonly obtained with current behaviour monitoring devices and passive monitoring systems that is not typically considered from an optimisation viewpoint, instead such system's practical usability is addressed more so than the data and its format for processing. Here the method of delivery of the data is addressed alongside the improvement of its usability in a monitoring application, with the developed schemes for fuzzy inference and fusion, dimension reduction and high dimension classification applicable cross-platform primarily in the intended field and any field which utilises extensive high dimension, nonlinear and complexly distributed input data as a means of inferring the state of a model-able system.

To keep abreast of all developments in practical behaviour monitoring is a difficult task, given that the state of the art is difficult to identify in each of the different sub-fields when each product released onto the market or researched in an academic environment utilises a different method of obtaining results from the previous development, or perhaps modifies a current standard without identifying similarities or differences with other such devices.

With this single application of each development in the field, rarely is it found that a technique is applied to another device or product in a similar context. The basis for many devices is typically a single premise or problem that has been previously addressed (detecting behaviours, analysing patterns etc.), yet the method used in solving the problem is less desirable than the creation of a new one designed specifically for the application at hand – i.e. the techniques designed for Verity are novel when compared with others due to the design requirements, yet there are many ways of reaching the same goal by study of other systems.

Working towards developing control methods for a real system has provided focus and direction to the research and informed the structure of the thesis into a document detailing its development through each stage, with every technique created and adapted with the Verity system in mind. Further stages of development as highlighted in the next section focus on the optimisation and enhancement of the techniques created within this thesis to better the implementation of Verity as a passive behaviour monitoring system for the elderly.

8.3 Further Work

As with any new development or modification of a technique, there can be identified areas of expansion to enhance suitability for an application. The research conducted here was certainly not exhaustive, however time, budget and experience constraints influenced the extent to which the research was conducted. In this section are detailed a number of possible areas of interest for further work on the project.

8.3.1 The Hidden Markov Model

The first stage of the research was in the development of the combined Hidden Markov Model and Fuzzy Inference System that would form the basis for all subsequent steps. The model operates effectively under the constraints of the system, with the variables and observations structured as they are. The inclusion of the Fuzzy Rule induction technique enhanced by the identified "least-rules" Unique Measure Merge condition means that over time, the Fuzzy Inference System can be updated to better attune to the user. However, it was only proposed that the standard Baum-Welch technique for HMM updating would be sufficient to ensure that all other bounds and parameters remained relevant to the user, and that the HMM and Fuzzy elements of the system would require independent updating in order to maintain adequate operation once re-combined.

A considerable and not unreasonable area identified for expansion arising from this part of the system is therefore in the updating of the models as a whole rather than as separate entities, foreseeably accomplishable through modification of the Baum-Welch algorithm to incorporate the parameter changes required for the linguistic rules to update periodically also; currently at the programming stage there needs to be an understanding of each observable state in order for the rules to make sense, and therefore training of the system is impossible without a knowledgeable operator in the initial stage.

Further enhancements can be made in the testing of the combined model with a wide variety of user data. In the research there was only a single set of data obtained with Verity that was used throughout, and informed of the format of the data such that "estimated" observations could be created to test the system further. With a larger number of subjects providing sensor data, the initial Fuzzy rules could be more generalised to a larger population, and through the induction method any irrelevant or erroneous parameters could be excluded as a means of tailoring more specifically to each user.

8.3.2 Dimension Reduction

Throughout the development of the CDALC constraints were identified that were thought might hinder the operation in the final hardware device. The Dijkstra graph traversal technique employed to link the prototypes of the data set is a search algorithm which takes an extensive amount of CPU power to reach a satisfactory solution for a large number of prototypes, and therefore on an embedded chip is quite possibly impractical when the data set to consider is one obtained from a lengthy period of operation. This drawback led to the notion that the dimension reduction scheme would not operate on-chip, but instead be handled by an offline system, or be utilised only in a period where the monitoring process could easily be relaxed in its intensity to allow for the on-board resources to focus on the prototype linking. As such, the further work on this element of the system would focus on the handling of the linking stage either through identification of another method or the development of a hardware system capable of operating at the required speed to integrate with the current Verity system.

A further possibility is to incorporate the Bloom Memory of the next stage into the dimension reduction technique to allow for a speed up in distance calculations – taking the locality expansion aspect of the memory as a means of providing the search algorithm with the distance from one prototype to the next. However, as in its basic form the Bloom Memory can be seen to replace the combined dimension reduction technique and neural network anyway, modifying the inferior technique for the purposes of classification is impractical unless the dimension reduction is required solely for the visualisation of state data and not its classification.

8.3.3 Bloom Memory

The theoretical operation of the Bloom Memory in its Hierarchical form is suitable for small, sparse data sets; in its classification of large, dense and high-resolution sets it requires more consideration of operational parameters when compared to the *k*NN technique it is intended to replace – despite enabling a better storage modality than simply recording all data points in a high-dimension space, denser sets require experimentation for the identification of optimum resolution values used to store the initial data where a *k*NN approach does not. The developed density calculations used to obtain a class probability are capable of reasoning the classification of a data point more adequately than with any other technique considered within the research however, as they base their result on the likelihood of the point being seen in the locality of other such points which share similar properties. It is with the previously mentioned dense sets however that problems may arise, but with further investigation the optimal radii and resolutions could be found for each set considered – and while not practical perhaps as an all-encompassing solution to the *k*NN method it certainly appears adequate enough for inclusion within the on-board system of Verity due to its rapid classification and storage space reduction.

The Localised Bloom Memory again operates similarly, however the ability to expand the storage space and modify the radius considered as “local” to a point is where the benefits lie during operation in Verity. As the user provides more and more data during standard operation, the system is expected to require further storage and faster identification of states given the larger amount of data to consider every time. Further work needs to focus on the expansion techniques and to assess at which point (if any) the memory fails to return adequate results due to an over-population of each class.

Testing both methods with the standard data sets used in the field does provide useful evidence to suggest further work is needed, yet for inclusion within Verity once again more data sets from testing with the actual hardware would be considered more beneficial.

8.3.4 Errors and Anomalies

It was identified in the early stages of the research those conditions under which Verity would alert a medical or care professional that the user was experiencing problems. The properties of the HMM as a probability model meant that intrinsically, detecting issues based on probabilities was more than adequate as it is all that is possible when states are identified based solely on observations; with some states also identified based on follow-up communication with the user, further anomalies and issues can be more adequately discovered and the probability based detections can act as backup reasoning for alerting of a problem.

It is desired that not only instantaneous errors be detected, but as previously stated: that long-term problems be identified as to the user's state and wellbeing. As with many applications detailed in the literature review, the overview of an entire series of states in comparison with perhaps another user or the same user at a different time may be an indicator that problems are developing that need addressing. Further work would focus on the assessment of which currently employed technique within the field is most appropriate for Verity without compromising any of the error types developed in this research. Once more, a larger set of data with which to test with would provide greater ability to tune the error detection methods specifically for the application.

8.3.5 General Improvements to the System

The Verity hardware was being developed simultaneously to the software algorithms and operational systems documented here. Each stage of the hardware development enabled a greater selection of possibilities for the system, with the incorporation of more sensors more states and more functions beyond the scope of the simple behaviour monitoring task could be identified and accomplished. Any further work on anything identified here as part of the methodology research into dealing with nonlinear and high dimensional sensor data obtained from behaviour monitoring devices would extend further than the requirements of Verity, theoretically improving its usability in a real environment beyond the considerations of the initial brief if they were implemented with appropriate consideration.

References

[1] The NHS Information Centre, Adult Social Care Statistics, "Community Care Statistics 2009-10: Social Services Activity Report, England," 978-1-84636-543-0, 2011.

[2] AgeUK, "Older people in the UK," Help the Aged 2008.

[3] M. Beckford. (8th Jan 2009). Number of elderly people receiving home care falls by 70000 after Labour reforms [Online]. Available: *http://www.telegraph.co.uk/news/uknews/4162856/Number-of-elderly-people-receiving-home-care-falls-by-70000-after-Labour-reforms.html* [Accessed: 9th March 2010].

[4] A. Hough. (19th Sept 2009). More families with elderly relatives forced to top up for old age care [Online]. Available: *http://www.telegraph.co.uk/health/healthnews/6204605/More-families-with-elderly-relatives-forced-to-top-up-for-old-age-care.html* [Accessed: 9th March 2010].

[5] CareTech. Emergency Monitoring Pendants [Online]. Available: *http://www.pendant.net.au/index.php* [Accessed: 20th Sept 2012].

[6] Care Innovations™. Care Innovations™ QuietCare® [Online]. Available: *http://www.careinnovations.com/products/quietcare-assisted-living-technology* [Accessed: 20th Sept 2012].

[7] Just Checking. Just Checking - Helping people to stay at home [Online]. Available: *http://www.justchecking.co.uk/* [Accessed: 20th Sept 2012].

[8] R. Kala, A. Shukla, and R. Tiwari, "Hybrid Intelligent Systems for Medical Diagnosis," in *Intelligent Medical Technologies and Biomedical Engineering: Tools and Applications*, vol.: IGI Global, 2010, pp. 187-202.

[9] G. Virone, M. Alwan, S. Dalal, S. W. Kell, B. Turner, J. A. Stankovic, and R. Felder, "Behavioral Patterns of Older Adults in Assisted Living," *Trans. Info. Tech. Biomed.*, vol. 12, no. 3, pp. 387-398, 2008.

[10] Y. Chen, D. Miao, and H. Zhang, "Neighborhood Outlier Detection," *Expert Systems with Applications*, vol. 37, no. 12, pp. 8745-8749, 2010.

[11] K. Z. Haigh, L. M. Kiff, and G. Ho, "The Independent LifeStyle Assistant: lessons learned," *Assist Technol*, vol. 18, no. 1, pp. 87-106, 2006.

[12] S. Junnila, I. Defee, M. Zakrzewski, A. M. Vainio, and J. Vanhala, "UUTE Home Network for Wireless Health Monitoring," in *Biocomputation, Bioinformatics, and Biomedical Technologies, 2008. BIOTECHNO '08. International Conference on*, 2008, pp. 125-130.

[13] P. Wolf, A. Schmidt, and M. Klein, "SOPRANO - An extensible, open AAL platform for elderly people based on semantical contracts " in *3rd Workshop on Artificial Intelligence Techniques for Ambient Intelligence (AITAml 08), 18th European Conference on Artificial Intelligence (ECAI 08)*, 2008.

[14] A. Sixsmith, S. Meuller, F. Lull, M. Klein, I. Bierhoff, S. Delaney, and R. Savage, M. Mokhtari, I. Khalil, J. Bauchet, D. Zhang, and C. Nugent "SOPRANO – An Ambient Assisted Living System for Supporting Older People at Home," in *Ambient Assistive Health and Wellness Management in the Heart of the City*, *Lecture Notes in Computer Science*, vol. 5597, M. Mokhtari, I. Khalil, J. Bauchet, D. Zhang, and C. Nugent, Eds.: Springer Berlin Heidelberg, 2009, pp. 233-236.

[15] F. Furfari and M.-R. Tazari, "Realizing Ambient Assisted Living Spaces with the PERSONA Platform," *ERCIM News*, vol. 2008, no. 74, 2008.

[16] M. Amoretti, S. Copelli, F. Wientapper, F. Furfari, S. Lenzi, and S. Chessa, "Sensor data fusion for activity monitoring in the PERSONA ambient assisted living project," *Journal of Ambient Intelligence and Humanized Computing*, vol. 4, no. 1, pp. 67-84, 2013.

[17] M.-R. Tazari, F. Furfari, J.-P. Ramos, and E. Ferro, H. Nakashima, H. Aghajan, and J. Augusto "The PERSONA Service Platform for AAL Spaces," in *Handbook of Ambient Intelligence and Smart Environments*, vol., H. Nakashima, H. Aghajan, and J. Augusto, Eds.: Springer US, 2010, pp. 1171-1199.

[18] vAssist. vAssist - Voice Controlled Assistive Care and Communication Services for the Home [Online]. Available: *http://vassist.cure.at/project_overview/* [Accessed: 9th Feb 2013].

[19] B. Krieg-Brückner, H. Bothmer, C. Budelmann, D. Crombie, A. Guerin, A. Heindorf, J. Lifante, A. B. Martínez, S. Millet, and E. Velleman. Assistance for Safe Mobility: the ASSAM Project [Online]. Available: *http://assam.nmshost.de/publications/* [Accessed: 9th Feb 2013].

[20] MobileSage Group AAL. MobileSage – Situated Adaptive Guidance for the Mobile Elderly [Online]. Available: *http://mobilesage.eu/* [Accessed: 9th Feb 2013].

[21] W. Burns, L. Chen, C. Nugent, M. Donnelly, K.-L. Skillen, and I. Solheim, F. Paternò, B. Ruyter, P. Markopoulos, C. Santoro, E. Loenen, and K. Luyten "A Conceptual Framework for Supporting Adaptive Personalized Help-on-Demand Services," in *Ambient*

Intelligence, *Lecture Notes in Computer Science*, vol. 7683, F. Paternò, B. Ruyter, P. Markopoulos, C. Santoro, E. Loenen, and K. Luyten, Eds.: Springer Berlin Heidelberg, 2012, pp. 427-432.

[22] A. Godfrey, R. Conway, D. Meagher, and G. ÓLaighin, "Direct measurement of human movement by accelerometry," *Medical Engineering & Physics*, vol. 30, no. 10, pp. 1364-1386, 2008.

[23] G. Virone, A. Wood, L. Selavo, Q. Cao, L. Fang, T. Doan, Z. He, R. Stoleru, S. Lin, and J. A. Stankovic, "An Assisted Living Oriented Information System Based on a Residential Wireless Sensor Network," in *Distributed Diagnosis and Home Healthcare, 2006. D2H2. 1st Transdisciplinary Conference on*, 2006, pp. 95-100.

[24] C. Chao and C. Pomalaza-Raez, "Design and Evaluation of a Wireless Body Sensor System for Smart Home Health Monitoring," in *Global Telecommunications Conference, 2009. GLOBECOM 2009. IEEE*, 2009, pp. 1-6.

[25] E. Campo and M. Chan, "Detecting abnormal behaviour by real-time monitoring of patients," in *In Proceedings of the AAAI-02 Workshop "Automation as Caregiver*, 2002, pp. 8-12.

[26] L. E. Baum and J. A. Eagon, "An inequality with applications to statistical estimation for probabilistic functions of Markov processes and to a model for ecology," *Bull. Amer. Math. Soc.*, vol. 3, no. 3, pp. 360-363, 1967.

[27] L. Baum and T. Petrie, "Statistical Inference for Probabilistic Functions of Finite State Markov Chains," *The Annals of Mathematical Statistics*, vol. 37, no. 6, pp. 1554-1563, 1966.

[28] M. Stanke and S. Waack, "Gene prediction with a hidden Markov model and a new intron submodel," *Bioinformatics*, vol. 19, no. suppl 2, pp. 215-225, 2003.

[29] V. De Fonzo, F. Aluffi-Pentini, and V. Parisi, "Hidden Markov Models in Bioinformatics," *Current Bioinformatics*, vol. 2, no. pp. 49-61, 2007.

[30] P. J. Green, R. Noad, and N. P. Smart, "Further Hidden Markov Model Cryptanalysis," in *7th International Conference on Cryptographic Hardware and Embedded Systems (CHES '05)*, 2005, pp. 61-74.

[31] C. Karlof and D. Wagner, "Hidden Markov Model Cryptanalysis," EECS Department, University of California, Berkeley UCB/CSD-03-1244, 2003.

[32] L. Satish and B. I. Gururaj, "Use of hidden Markov models for partial discharge pattern classification," *Electrical Insulation, IEEE Transactions on*, vol. 28, no. 2, pp. 172-182, 1993.

[33] P.-C. Chung and C.-D. Liu, "A daily behavior enabled hidden Markov model for human behavior understanding," *Pattern Recognition*, vol. 41, no. 5, pp. 1572-1580, 2008.

[34] D. Kosmopoulos, P. Antonakaki, K. Valasoulis, and D. Katsoulas, "Monitoring human behavior in an assistive environment using multiple views," in *1st International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '08)*, 2008, pp. 1-6.

[35] N. Oliver, E. Horvitz, and A. Garg, "Layered Representations for Human Activity Recognition," in *4th IEEE International Conference on Multimodal Interfaces (ICMI 2002)*, 2002, pp. 3-8.

[36] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," in *Readings in speech recognition*, vol.: Morgan Kaufmann Publishers Inc., 1990, pp. 267-296.

[37] L. Rabiner and B. Juang, "An introduction to hidden Markov models," *ASSP Magazine, IEEE*, vol. 3, no. 1, pp. 4-16, 1986.

[38] Y. Bengio, "Markovian models for sequential data," *Neural Computing Surveys*, no. 2, pp. 129--162, 1999.

[39] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *Information Theory, IEEE Transactions on*, vol. 13, no. 2, pp. 260-269, 1967.

[40] Y. Li, "Hidden Markov models with states depending on observations," *Pattern Recognition Letters*, vol. 26, no. 7, pp. 977-984, 2005.

[41] L. Atallah and G.-Z. Yang, "The use of pervasive sensing for behaviour profiling -- a survey," *Pervasive and Mobile Computing*, vol. 5, no. 5, pp. 447-464, 2009.

[42] L. Atallah, M. ElHelw, J. Pansiot, D. Stoyanov, L. Wang, B. Lo, and G. Z. Yang, S. Leonhardt, T. Falck, and P. Mähönen "Behaviour Profiling with Ambient and Wearable Sensing," in *4th International Workshop on Wearable and Implantable Body Sensor Networks (BSN 2007)*, *IFMBE Proceedings*, vol. 13, S. Leonhardt, T. Falck, and P. Mähönen, Eds.: Springer Berlin Heidelberg, 2007, pp. 133-138.

[43] H. Lee, K. Park, B. Lee, J. Choi, and R. Elmasri, "Issues in data fusion for healthcare monitoring," in *1st International Conference on PErvasive Technologies Related to Assistive Environments (PETRA '08)*, 2008, pp. 1-8.

[44] M. Dong and D. He, "Hidden semi-Markov model-based methodology for multi-sensor equipment health diagnosis and prognosis," *European Journal of Operational Research*, vol. 178, no. 3, pp. 858-878, 2007.

[45] L. Atallah, B. Lo, Y. Guang-Zhong, and F. Siegemund, "Wirelessly Accessible Sensor Populations (WASP) for Elderly Care Monitoring,"

in *Second International Conference on Pervasive Computing Technologies for Healthcare*, 2008, pp. 2--7.

[46] H. B. Mitchell, *Multi-sensor Data Fusion: An Introduction*: Springer London, Limited, 2007.

[47] A. Makarenko, A. Brooks, T. Kaupp, H. Durrant-Whyte, and F. Dellaert, "Decentralised data fusion: A graphical model approach," in *Information Fusion, 2009. FUSION '09. 12th International Conference on*, 2009, pp. 545-554.

[48] D. M. Buede and P. Girardi, "A target identification comparison of Bayesian and Dempster-Shafer multisensor fusion," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 27, no. 5, pp. 569-577, 1997.

[49] S. Cooper and H. Durrant-Whyte, "A Kalman filter model for GPS navigation of land vehicles," in *Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on*, 1994, vol. 1, pp. 157-163.

[50] J. Z. Sasiadek and Q. Wang, "Sensor Fusion Based on Fuzzy Kalman Filtering for Autonomous Robot Vehicle," in *1999 IEEE International Conference on Robotics and Automation*, 1999, vol. 4, pp. 2970-2975.

[51] D. Simon, "Kalman filtering with state constraints: a survey of linear and nonlinear algorithms," *Control Theory & Applications, IET*, vol. 4, no. 8, pp. 1303-1318, 2010.

[52] D. L. Hall and J. Llinas, "An introduction to multisensor data fusion," *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6-23, 2002.

[53] C. Aliustaoglu, H. M. Ertunc, and H. Ocak, "Tool wear condition monitoring using a sensor fusion model based on fuzzy inference system," *Mechanical Systems and Signal Processing*, vol. 23, no. 2, pp. 539-546, 2009.

[54] R. Morales-Menendez, A. J. Vallejo, J. A. Nolazco-Flores, and P. Garcia-Perera, "Low-cost cutting tool diagnosis based on sensor-fusion," in *IFAC Conference on Cost Effective Automation in Networked Product Development and Manufacturing*, 2007, vol. 1, pp. 141-146.

[55] S.-H. Hwang, "A Triadic Approach of Hierarchical Classes Analysis on Folksonomy Mining," *International Journal of Computer Science and Network Security*, vol. 7, no. 8, 2007.

[56] K. Bernardin, K. Ogawara, K. Ikeuchi, and R. Dillmann, "A sensor fusion approach for recognizing continuous human grasping sequences using hidden Markov models," *Robotics, IEEE Transactions on*, vol. 21, no. 1, pp. 47-57, 2005.

[57] L. Baum, T. Petrie, G. Soules, and N. Weiss, "A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains," *The Annals of Mathematical Statistics*, vol. 41, no. 1, pp. 164-171, 1970.

[58] Z. Chun and S. Weihua, "Wearable Sensor-Based Hand Gesture and Daily Activity Recognition for Robot-Assisted Living," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 41, no. 3, pp. 569-573, 2011.

[59] D. Curone, G. M. Bertolotti, A. Cristiani, E. L. Secco, and G. Magenes, "A Real-Time and Self-Calibrating Algorithm Based on Triaxial Accelerometer Signals for the Detection of Human Posture and Activity," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 14, no. 4, pp. 1098-1105, 2010.

[60] A. Pantelopoulos and N. G. Bourbakis, "A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 40, no. 1, pp. 1-12, 2010.

[61] C. Jia-Ren Chang and T. Cheng-Chi, "A New Wireless-Type Physiological Signal Measuring System Using a PDA and the Bluetooth Technology," in *Industrial Technology, 2006. ICIT 2006. IEEE International Conference on*, 2006, pp. 3026-3031.

[62] U. Maurer, A. Rowe, A. Smailagic, and D. P. Siewiorek, "eWatch: A Wearable Sensor and Notification Platform," in *International Workshop on Wearable and Implantable Body Sensor Networks (BSN '06)*, 2006, pp. 142-145.

[63] S. Das and D. Cook, J, D. Zhang and M. Mokhtari "Health Monitoring in an agent-based smart home by activity prediction," in *Toward a human-friendly assistive environment : ICOST '2004, 2nd International Conference on Smart Homes and Health Telematics*, *Assistive technology research series*, vol. 14, D. Zhang and M. Mokhtari, Eds. Amsterdam ; Washington, DC: IOS Press, 2004, pp. 3-14.

[64] T. S. Barger, D. E. Brown, and M. Alwan, "Health-status monitoring through analysis of behavioural patterns," *Systems, Man and Cybernetics, Part A, IEEE Transactions on*, vol. 35, no. 1, pp. 22-27, 2005.

[65] T. Tamura, T. Togawa, M. Ogawa, and M. Yoda, "Fully automated health monitoring system in the home," *Medical Engineering & Physics*, vol. 20, no. 8, pp. 573-579, 1998.

[66] Y. Hairong, H. Hongwei, X. Youzhi, and M. Gidlund, "Wireless sensor network based E-health system: implementation and experimental results," *Consumer Electronics, IEEE Transactions on*, vol. 56, no. 4, pp. 2288-2295, 2010.

[67] L. Hyun, C. Jae Sung, and R. Elmasri, "A Static Evidential Network for Context Reasoning in Home-Based Care," *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, vol. 40, no. 6, pp. 1232-1243, 2010.

[68] S. Patel, K. Lorincz, R. Hughes, N. Huggins, J. Growdon, D. Standaert, M. Akay, J. Dy, M. Welsh, and P. Bonato, "Monitoring Motor Fluctuations in Patients With Parkinson's Disease Using Wearable Sensors," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 13, no. 6, pp. 864-873, 2009.

[69] A. Pentland, "Healthwear: medical technology becomes wearable," *Computer*, vol. 37, no. 5, pp. 42-49, 2004.

[70] T. Xiao-Fei, Z. Yuan-Ting, C. C. Y. Poon, and P. Bonato, "Wearable Medical Systems for p-Health," *IEEE Reviews in Biomedical Engineering*, no. pp. 62-74, 2008.

[71] P. Bonato, "Advances in wearable technology and its medical applications," in *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, 2010, pp. 2021-2024.

[72] R. DeVaul, M. Sung, J. Gips, and A. Pentland, "MIThril 2003: Applications and Architecture," in *7th IEEE International Symposium on Wearable Computers*, 2003, pp. 4-11.

[73] C. Sopavanit, T. Desudchit, and P. Riyamongkol, "Wireless and wearable EKG device with lossless compression for on-line post-surgery heart monitoring system," in *Robotics and Biomimetics, 2008. ROBIO 2008. IEEE International Conference on*, 2009, pp. 1218-1223.

[74] V. Mukala, V. Lakafosis, A. Traille, and M. M. Tentzeris, "A novel Zigbee-based low-cost, low-power wireless EKG system," in *2010 IEEE MTT-S International Microwave Symposium Digest (MTT)*, 2010, pp. 624-627.

[75] T. R. F. Fulford-Jones, W. Gu-Yeon, and M. Welsh, "A portable, low-power, wireless two-lead EKG system," in *Engineering in Medicine and Biology Society, 2004. IEMBS '04. 26th Annual International Conference of the IEEE*, 2004, vol. 1, pp. 2141-2144.

[76] R. Ogawa and T. Togawa, "Attempts at monitoring health status in the home," in *1st Annual International Conference On Microtechnologies in Medicine and Biology*, 2000, pp. 552-556.

[77] G. Lin, T. Nakajima, P. Rahul, and A. Hodge, "Seamlessly Embedded Heart Rate Monitor," US 2010/0113950 A1, 2010.

[78] B. Lo, S. Thiemjarus, R. King, and G. Yang, "Body Sensor Network - A Wireless Sensor Platform for Pervasive Healthcare Monitoring," in

Adjunct Proceedings of the 3rd International conference on Pervasive Computing 2005, pp. 77-80.

[79] F. Zhou, J. Jiao, S. Chen, and D. Zhang, "A Case-Driven Ambient Intelligence System for Elderly in-Home Assistance Applications," *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, vol. 41, no. 2, pp. 179-189, 2011.

[80] J. Yao, R. Schmitz, and S. Warren, "A wearable point-of-care system for home use that incorporates plug-and-play and wireless standards," *Trans. Info. Tech. Biomed.*, vol. 9, no. 3, pp. 363-371, 2005.

[81] A. Tablado, A. Illarramendi, J. bermudez, and A. Goni, "Intelligent monitoring of elderly people," in *Information Technology Applications in Biomedicine, 2003. 4th International IEEE EMBS Special Topic Conference on*, 2003, pp. 78-81.

[82] C. C. Aggarwal and P. S. Yu, "Outlier detection for high dimensional data," in *in 'ACM SIGMOD International Conference on Management of Data', ACM*: Press, 2001, pp. 37--46.

[83] E. Fix and J. L. Hodges, "Discriminatory analysis, nonparametric discrimination: Consistency properties," *US Air Force School of Aviation Medicine*, vol. Technical Report 4, no. 3, p. 477, 1951.

[84] H.-P. Kriegel, P. Kröger, and A. Zimek, "Detecting clusters in moderate-to-high dimensional data: subspace clustering, pattern-based clustering, and correlation clustering," *Proc. VLDB Endow.*, vol. 1, no. 2, pp. 1528-1529, 2008.

[85] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, "When is "Nearest Neighbor" Meaningful?," in *7th International Conference on Database Theory (ICDT '99)*, 1999, pp. 217-235.

[86] A. Hinneburg, C. C. Aggarwal, and D. A. Keim, "What is the Nearest Neighbor in High Dimensional Spaces?," in *26th International Conference on Very Large Data Bases (VLDB '00)*, 2000, pp. 506-515.

[87] J. W. Sammon, Jr., "A Nonlinear Mapping for Data Structure Analysis," *Computers, IEEE Transactions on*, vol. C-18, no. 5, pp. 401-409, 1969.

[88] J. Tenenbaum, V. de Silva, and J. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, no. 5500, pp. 2319-2323, 2000.

[89] J. A. Lee, A. Lendasse, and M. Verleysen, "Nonlinear projection with curvilinear distances: Isomap versus curvilinear distance analysis," *Neurocomputing*, vol. 57, no. pp. 49-76, 2004.

[90] J. Lee, A. Lendasse, and M. Verleysen, "A robust nonlinear projection method," in *In Proceedings of ESANN'2000, Belgium*, 2000, pp. 13--20.

[91] P. Demartines and J. Herault, "Curvilinear component analysis: a self-organizing neural network for nonlinear mapping of data sets," *Neural Networks, IEEE Transactions on*, vol. 8, no. 1, pp. 148-154, 1997.

[92] I. Fodor. A Survey of Dimension Reduction Techniques [Online]. Available: *http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.8.5098* [Accessed: 9th Feb 2013].

[93] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 6, pp. 559-572, 1901.

[94] D. X. Sun, "Feature dimension reduction using reduced-rank maximum likelihood estimation for hidden Markov models," in *4th International Conference on Spoken Language, 1996 (ICSLP '96)*, 1996, vol. 1, pp. 244-247.

[95] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-Based Learning Algorithms," *Mach. Learn.*, vol. 6, no. 1, pp. 37-66, 1991.

[96] T. Cover and P. Hart, "Nearest neighbor pattern classification," *Information Theory, IEEE Transactions on*, vol. 13, no. 1, pp. 21-27, 1967.

[97] J. L. Bentley, "Survey of techniques for fixed radius near neighbor searching," Stanford University, Stanford, CA 1975.

[98] J. Beringer and E. Hüllermeier, "An efficient algorithm for instance-based learning on data streams," in *7th Industrial Conference on Advances in Data Mining: Theoretical Aspects and Applications (ICDM '07)*, 2007, pp. 34-48.

[99] C. Li and G. Biswas, "Finding Behavior Patterns from Temporal Data using Hidden Markov Model based Unsupervised Classification," *Computer Science Department*. Nashville: Vanderbilt University.

[100] B. T. Fine, "Unsupervised anomaly detection with minimal sensing," in *47th Annual Southeast Regional Conference*, 2009, pp. 1-5.

[101] M. Burgess, "Probabilistic anomaly detection in distributed computer networks," *Science of Computer Programming*, vol. 60, no. 1, pp. 1-26, 2006.

[102] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 1-58, 2009.

[103] S. Kim, N. W. Cho, B. Kang, and S.-H. Kang, "Fast outlier detection for very large log data," *Expert Systems with Applications*, vol. 38, no. 8, pp. 9587-9596, 2011.

[104] N. Ye, "A markov chain model of temporal behavior for anomaly detection," in *Workshop on Information Assurance and Security*, 2000.

[105] B. K. L. Fei, J. H. P. Eloff, M. S. Olivier, and H. S. Venter, "The use of self-organising maps for anomalous behaviour detection in a digital investigation," *Forensic Science International*, vol. 162, no. 1-3, pp. 33-37, 2006.

[106] K.-j. Kim, "Artificial neural networks with evolutionary instance selection for financial forecasting," *Expert Syst. Appl.*, vol. 30, no. 3, pp. 519-526, 2006.

[107] D. LeRoux, S.-H. Chen, P. Wang, and T.-W. Kuo "Comparison of Instance-Based Techniques for Learning to Predict Changes in Stock Prices," in *Computational Intelligence in Economics and Finance*, vol., S.-H. Chen, P. Wang, and T.-W. Kuo, Eds.: Springer Berlin Heidelberg, 2007, pp. 135-143.

[108] G. Góra and A. Wojna, "RIONA: A New Classification System Combining Rule Induction and Instance-Based Learning," *Fundam. Inf.*, vol. 51, no. 4, pp. 369-390, 2002.

[109] J. Toyama, M. Kudo, and H. Imai, "Probably correct k-nearest neighbor search in high dimensions," *Pattern Recognition*, vol. 43, no. 4, pp. 1361-1372, 2010.

[110] P. Indyk and R. Motwani, "Approximate nearest neighbors: towards removing the curse of dimensionality," in *30th Annual ACM symposium on Theory of Computing*, 1998, pp. 604-613.

[111] M. Slaney and M. Casey, "Locality-Sensitive Hashing for Finding Nearest Neighbors [Lecture Notes]," *Signal Processing Magazine, IEEE*, vol. 25, no. 2, pp. 128-131, 2008.

[112] A. Andoni and P. Indyk, "Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions," *Commun. ACM*, vol. 51, no. 1, pp. 117-122, 2008.

[113] B. H. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Commun. ACM*, vol. 13, no. 7, pp. 422-426, 1970.

[114] D. E. Knuth, *The art of computer programming, volume 2 (3rd ed.): seminumerical algorithms*: Addison-Wesley Longman Publishing Co., Inc., 1997.

[115] D. Guo, J. Wu, H. Chen, and X. Luo, "Theory and Network Applications of Dynamic Bloom Filters," in *INFOCOM 2006. 25th IEEE*

International Conference on Computer Communications. Proceedings, 2006, pp. 1-12.

[116] J. Aguilar-Saborit, P. Trancoso, V. Muntes-Mulero, and J. L. Larriba-Pey, "Dynamic count filters," *SIGMOD Rec.*, vol. 35, no. 1, pp. 26-32, 2006.

[117] M. R. Gorai, K. S. Sridharan, T. Aditya, R. Mukkamala, and S. Nukavarapu, "Employing bloom filters for privacy preserving distributed collaborative kNN classification," in *Information and Communication Technologies (WICT), 2011 World Congress on*, 2011, pp. 495-500.

[118] L. Fan, P. Cao, J. Almeida, and A. Z. Broder, "Summary cache: a scalable wide-area web cache sharing protocol," *IEEE/ACM Trans. Netw.*, vol. 8, no. 3, pp. 281-293, 2000.

[119] M. Mitzenmacher, "Compressed Bloom filters," *Networking, IEEE/ACM Transactions on*, vol. 10, no. 5, pp. 604-612, 2002.

[120] A. Kirsch and M. Mitzenmacher, "Distance-Sensitive Bloom Filters," in *Eighth Workshop on Algorithm Engineering and Experiments (ALENEX)*, 2006.

[121] G. Virone, N. Noury, and J. Demongeot, "A system for automatic measurement of circadian activity deviation in telemedicine," *IEEE Transactions on Biomedical Engineering*, no. 49, pp. 1463-1469, 2002.

[122] A. Helal, D. J. Cook, and M. Schmalz, "Smart Home-Based Health Platform for Behavioral Monitoring and Alteration of Diabetes Patients," *Journal of Diabetes Science and Technology*, vol. 3, no. 1, pp. 141-148, 2009.

[123] L. E. Krahn and H. L. Gonzalez-Arriaza, "Narcolepsy With Cataplexy," *Am J Psychiatry*, vol. 161, no. 12, pp. 2181-2184, 2004.

[124] S. Simani, C. Fantuzzi, and R. Patton, *Model-based fault diagnosis in dynamic systems using identification techniques*. New York: Springer, 2002.

[125] H. Junker, O. Amft, P. Lukowicz, and G. Tröster, "Gesture spotting with body-worn inertial sensors to detect user activities," *Pattern Recognition*, vol. 41, no. 6, pp. 2010-2024, 2008.

[126] C. H. Griffin, "Behavior Detection using Confidence Intervals of Hidden Markov Models," *Journal Name: IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1484-1492, 2009.

[127] M. F. Abbod, D. G. von Keyserlingk, D. A. Linkens, and M. Mahfouf, "Survey of utilisation of fuzzy technology in Medicine and Healthcare," *Fuzzy Sets and Systems*, vol. 120, no. 2, pp. 331-349, 2001.

[128] S. S. Joshi and V. V. Phoha, "Investigating hidden Markov models capabilities in anomaly detection," in *43rd Annual Southeast Regional Conference*, 2005, vol. 1, pp. 98-103.

[129] V. Jecheva, "About Some Applications of Hidden Markov Model in IntrusionDetection Systems," in *International Conference on Computer Systems and Technologies (CompSysTech '06)*, 2006.

[130] Toumaz Healthcare Ltd. AMx™ Mixed Signal Technology [Online]. Available: *http://www.toumaz.com/page.php?page=amx* [Accessed: 9th Feb 2013].

[131] O. Omeni, A. Wong, A. J. Burdett, and C. Toumazou, "Energy Efficient Medium Access Protocol for Wireless Medical Body Area Sensor Networks," *Biomedical Circuits and Systems, IEEE Transactions on*, vol. 2, no. 4, pp. 251-259, 2008.

[132] A. Voss, "Longitudinal analysis of heart rate variability," *Journal of Electrocardiology*, vol. 40, no. 1, Supplement 1, pp. S26-S29, 2007.

[133] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME – Journal of Basic Engineering*, no. 82 (Series D), pp. 35-45, 1960.

[134] B. van der Pol and J. van der Mark, "The heartbeat considered as a relaxation oscillation, and an electrical model of the heart," in *Philosophical Magazine*, 1928, pp. 763–775.

[135] L. H. van der Tweel, F. L. Meijler, and F. J. van Capelle, "Synchronization of the heart," *Journal of Applied Physiology*, vol. 34, no. 2, pp. 283-287, 1973.

[136] H. Tasaki, T. Serita, A. Irita, O. Hano, I. Iliev, C. Ueyama, K. Kitano, S. Seto, M. Hayano, and K. Yano, "A 15-Year Longitudinal Follow-Up Study of Heart Rate and Heart Rate Variability in Healthy Elderly Persons," *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences*, vol. 55, no. 12, pp. M744-M749, 2000.

[137] P. Nickel and F. Nachreiner, "Sensitivity and Diagnosticity of the 0.1-Hz Component of Heart Rate Variability as an Indicator of Mental Workload," *Human Factors: The Journal of the Human Factors and Ergonomics Society*, vol. 45, no. 4, pp. 575-590, 2003.

[138] H.-W. Lee, J.-W. Lee, W.-G. Jung, and G.-K. Lee, "The Periodic Moving Average Filter for Removing MotionArtifacts from PPG Signals," *International Journal of Control, Automation, and Systems*, vol. 5, no. 6, pp. 701-706, 2009.

[139] R. H. Brown, "The Piezo Solution for Vital Signs Monitoring," in *Medical Design Technology*, vol. 12, March 2008, pp. 36.

[140] J. D. Bryars and D. Cavanaugh, "Heart pulse monitor," 5807267, 1998.

[141] M. Kelarestaghi, M. Slimane, and N. Vincent, "Introduction of fuzzy logic in the Hidden Markov Models," in *2nd International Conference in Fuzzy Logic and Technology*, 2001, pp. 14-16.

[142] N. K. Verma and M. Hanmandlu, "Additive and Non-Additive Fuzzy Hidden Markov Models," *Fuzzy Systems, IEEE Transactions on*, vol. 18, no. 1, pp. 40-56, 2009.

[143] R. M. de Moraes and L. dos Santos Machado, "Using fuzzy hidden Markov models for online training evaluation and classification in virtual reality simulators," *International Journal of General Systems*, vol. 33, no. 2, pp. 281 - 288, 2004.

[144] E. H. Mamdani and S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller," *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1-13, 1975.

[145] T.-P. Hong and C.-Y. Lee, "Induction of fuzzy rules and membership functions from training examples," *Fuzzy Sets and Systems*, vol. 84, no. 1, pp. 33-47, 1996.

[146] S. Nefti, M. Oussalah, and U. Kaymak, "A New Fuzzy Set Merging Technique Using Inclusion-Based Fuzzy Clustering," *Fuzzy Systems, IEEE Transactions on*, vol. 16, no. 1, pp. 145-161, 2008.

[147] A. M. Abhilash, B. Yann-Aël, and B. Gianluca, "New Routes from Minimal Approximation Error to Principal Components," *Neural Process. Lett.*, vol. 27, no. 3, pp. 197-207, 2008.

[148] D. Meng, Y. Leung, and Z. Xu, "Passage method for nonlinear dimensionality reduction of data on multi-cluster manifolds," *Pattern Recognition*, vol. 46, no. 8, pp. 2175-2186, 2013.

[149] G. Rosman, M. Bronstein, A. Bronstein, and R. Kimmel, "Nonlinear Dimensionality Reduction by Topologically Constrained Isometric Embedding," *International Journal of Computer Vision*, vol. 89, no. 1, pp. 56-68, 2010.

[150] M. Katajamaa, J. Miettinen, and M. Orešič, "MZmine: toolbox for processing and visualization of mass spectrometry based molecular profile data," *Bioinformatics*, vol. 22, no. 5, pp. 634-636, 2006.

[151] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269-271, 1959.

[152] A. Frank and A. Asuncion. UCI Machine Learning Repository [Online]. Available: *http://archive.ics.uci.edu/ml* [Accessed: 29th Dec 2010].

[153] S. Tarkoma, C. E. Rothenberg, and E. Lagerspetz, "Theory and Practice of Bloom Filters for Distributed Systems," *Communications Surveys & Tutorials, IEEE*, vol. 14, no. 1, pp. 131-155, 2012.

[154] P. W. Frey and D. J. Slate, "Letter Recognition Using Holland-Style Adaptive Classifiers," *Mach. Learn.*, vol. 6, no. 2, pp. 161-182, 1991.

[155] C. Plant, C. Böhm, B. Tilg, and C. Baumgartner, "Enhancing instance-based classification with local density: a new algorithm for classifying unbalanced biomedical data," *Bioinformatics*, vol. 22, no. 8, pp. 981-988, 2006.

[156] Q. Lv, W. Josephson, Z. Wang, M. Charikar, and K. Li, "Multi-probe LSH: efficient indexing for high-dimensional similarity search," in *33rd International Conference on Very Large Data Bases (VLDB '07)*, 2007, pp. 950-961.

[157] H. Yu, X. Bin, B. Veeravalli, and F. Dan, "Locality-Sensitive Bloom Filter for Approximate Membership Query," *Computers, IEEE Transactions on*, vol. 61, no. 6, pp. 817-830, 2012.

[158] M. Sugiura, Y. Sassa, H. Jeong, N. Miura, Y. Akitsuki, K. Horie, S. Sato, and R. Kawashima, "Multiple brain networks for visual self-recognition with different sensitivity for motion and body part," *NeuroImage*, vol. 32, no. 4, pp. 1905-1917, 2006.

[159] J. Blumberg and G. Kreiman, "How cortical neurons help us see: visual recognition in the human brain," *The Journal of Clinical Investigation*, vol. 120, no. 9, pp. 3054-3063, 2010.

[160] E. D. Grossman and R. Blake, "Brain Areas Active during Visual Perception of Biological Motion," *Neuron*, vol. 35, no. 6, pp. 1167-1175, 2002.

[161] A. Nestor, D. C. Plaut, and M. Behrmann, "Unraveling the distributed neural code of facial identity through spatiotemporal pattern analysis," *Proceedings of the National Academy of Sciences*, vol. 108, no. 24, pp. 9998-10003, 2011.

[162] V. Bruce and A. Young, "Understanding face recognition," *British Journal of Psychology*, vol. 77, no. 3, pp. 305-327, 1986.

[163] D. Marr, S. Ullman, and T. Poggio, *Vision: A Computational Investigation Into the Human Representation and Processing of Visual Information*: MIT Press, 2010.

[164] G. Van Belle, P. De Graef, K. Verfaillie, T. Busigny, and B. Rossion, "Whole not hole: Expert face recognition requires holistic perception," *Neuropsychologia*, vol. 48, no. 9, pp. 2620-2629, 2010.

[165] L. D. Harmon and B. Julesz, "Masking in Visual Recognition: Effects of Two-Dimensional Filtered Noise," *Science*, vol. 180, no. 4091, pp. 1194-1197, 1973.

[166] A. W. Yip and P. Sinha, "Contribution of color to face recognition," *Perception*, vol. 31, no. 8, pp. 995-1003, 2002.

[167] A. M. Burton, S. Wilson, M. Cowan, and V. Bruce, "Face Recognition in Poor-Quality Video: Evidence From Security Surveillance," *Psychological Science*, vol. 10, no. 3, pp. 243-248, 1999.

[168] J. Ying, T. Kirubarajan, K. R. Pattipati, and A. Patterson-Hine, "A hidden Markov model-based algorithm for fault diagnosis with partial and imperfect tests," *Systems, Man and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 30, no. 4, pp. 463-473, 2000.

[169] S. Ding, *Model-based fault diagnosis techniques : design schemes, algorithms and tools*. Berlin: Springer, 2008.

[170] R. Patton, P. Frank, and R. Clark, *Fault diagnosis in dynamic systems*. New York, London: Prentice-Hall, 1989.

Appendix A

The traditional HMM consists of five key components. There are N states (S) into which M observations (V) can belong, with probabilities defined by a probability distribution $B = b_j(k)$, where j is the current state and k the observation number. The probability of transitioning from one state to another is an element in a state transition probability distribution matrix which is defined as $A = a_{ij}$ where i is the current state and j the proceeding state (i.e. the state after the transition). The final element of the model is termed π: the initial state distribution, which is a record of the probability of seeing any state at the first time instance. The elements of the model are defined thus:

$$S = \{S_1, S_2, S_3, \ldots, S_N\} \tag{A1}$$

$$V = \{V_1, V_2, V_3, \ldots, V_M\} \tag{A2}$$

$$a_{ij} = P\left(q_{t+1} = S_j \middle| q_t = S_i\right) \tag{A3}$$

(Note that q_t is the state at time t)

$$b_j(k) = P\left(V_k \text{ at } t \middle| q_t = S_j\right) \qquad \begin{matrix} 1 \le j < N \\ 1 \le k < M \end{matrix} \tag{A4}$$

$$\pi_i = P(q_1 = S_i) \qquad 1 \le j < N \tag{A5}$$

The model is commonly denoted in its compact form as:

$$\lambda = (A, B, \pi) \tag{A6}$$

A Hidden Markov Model can solve 2 problems:

1. Given an observation sequence $O = [O_1, O_2, \ldots, O_T]$ and a model λ, how can the probability of the observation sequence's occurrence, $P(O|\lambda)$ be efficiently calculated?
2. Given an observation sequence $O = [O_1, O_2, \ldots, O_T]$ and a model λ, how can an optimal state sequence $Q = [q_1, q_2, \ldots, q_T]$ be chosen to best explain the observations?

Calculating sequence probability

For the solution to both problems, the Forward-Backward procedure can be used, with the forward part being solely of use for the first problem:

$$\alpha_t(i) = P(O_1, O_2, \ldots, O_t, q_t = S_i | \lambda) \qquad \text{(A7)}$$

I. Initialise:

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \le i < N \qquad \text{(A8)}$$

II. Inductive step:

$$\alpha_{t+1}(j) = \left[\sum_{i=i}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad \begin{matrix} 1 \le t < T-1 \\ 1 \le j < N \end{matrix} \qquad \text{(A9)}$$

III. Terminate:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \qquad \text{(A10)}$$

Thus by induction can be found the probability of terminating in state S_i at time t having been presented with the observation sequence and the model.

Calculating an optimal sequence

With the introduction of a state sequence requirement, the probabilities of proceeding states to the end of the sequence must be taken into account. For this problem the backward part of the Forward-Backward procedure is calculated:

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, \ldots, O_T | q_t = S_i, \lambda) \qquad \text{(A11)}$$

I. Arbitrary Initialisation - assuming that the end state is certain, the previous probabilities of a state sequence occurring can be calculated:

$$\beta_T(i) = 1 \qquad 1 \le i < N \qquad \text{(A12)}$$

II. Inductive step:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \qquad \begin{matrix} t = T-1, T-2, \ldots, 1 \\ 1 \le i < N \end{matrix} \qquad \text{(A13)}$$

This calculation aids in the finding of an optimal state sequence for the given observations, yet the definition of “optimal” is open to interpretation. A state sequence may consist of states which are most likely at each time step given the observation sequence - regardless of the possibility of the state sequence occurring. It may also be a sequence which most logically flows from one state to the next, i.e. takes into account the probability of transitioning from the previous state to the current, along with the observation sequence. For the first of such variations, both previously calculated Forward-Backward parts are employed together.

The probability of being in a single state at a time, given the observations and model is defined as:

$$\gamma_t(i) = P(q_t = S_i | O, \lambda) \tag{A14}$$

The above equation can be written in terms of the Forward-Backward variables:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{A15}$$

Taking the maximum value of (A14) gives the individually most likely state at that time:

$$q_t = \underset{1 \le i \le N}{\arg\max} \left[\gamma_t(i)\right] \qquad 1 \le t < T \tag{A16}$$

The Viterbi Algorithm takes into account the likelihood of state transitions in sequence, unlike the previous method. In this property it can be seen to have globally optimised the output, using all available information from within the model. Therefore the resulting state sequence is entirely possible given the observations presented to the model. However, the algorithm adjusts the entire sequence to match the most likely state at the time; if the next observation most likely belongs to a state which it is unlikely to reach from the current state, the backtracked sequence may change to accommodate it and increase the likelihood of the sequence.

What is being determined can be expressed as $P(Q,O|\lambda)$: the probability of seeing the state sequence *and* the observation sequence given the model. $\delta_t(i)$ is the highest probability along a single state sequence as calculated at time t, accounting for the first t observations and terminating with state S_j. The state sequence itself is given in the array ψ, which is populated with the state maximising that probability calculated by δ at each step.

I. Initialise:

$$\delta_1(i) = \pi_i b_i(O_1) \qquad 1 \le i < N \qquad \text{(A17)}$$

$$\psi_1(i) = 0 \qquad \text{(A18)}$$

II. Recursion Step:

$$\delta_t(j) = \max_{1 \le i \le N}\left[\delta_{t-1}(i) a_{ij}\right] b_j(O_t) \qquad \begin{matrix} 2 \le t < T \\ 1 \le j < N \end{matrix} \qquad \text{(A19)}$$

$$\psi_t(i) = \arg\max_{1 \le i \le N}\left[\delta_{t-1}(i) a_{ij}\right] \qquad \begin{matrix} 2 \le t < T \\ 1 \le j < N \end{matrix} \qquad \text{(A20)}$$

III. Terminate:

$$P^* = \max_{1 \le i \le N}\left[\delta_T(i)\right] \qquad \text{(A21104)}$$

$$q_T^* = \arg\max_{1 \le i \le N}\left[\delta_T(i)\right] \qquad \text{(A22)}$$

IV. The backtracking procedure:

$$P^* = \max_{1 \le i \le N}\left[\delta_T(i)\right] \qquad \text{(A23)}$$

$$q_t^* = \psi_{t+1}\left(q_{t+1}^*\right) \qquad t = T-1, T-2, \ldots, 1 \qquad \text{(A24)}$$

The resulting state sequence, ψ, is that which is most likely to have occurred in order to reach the state most likely to have emitted the observation at time T, given transitions from previous states.

Appendix B

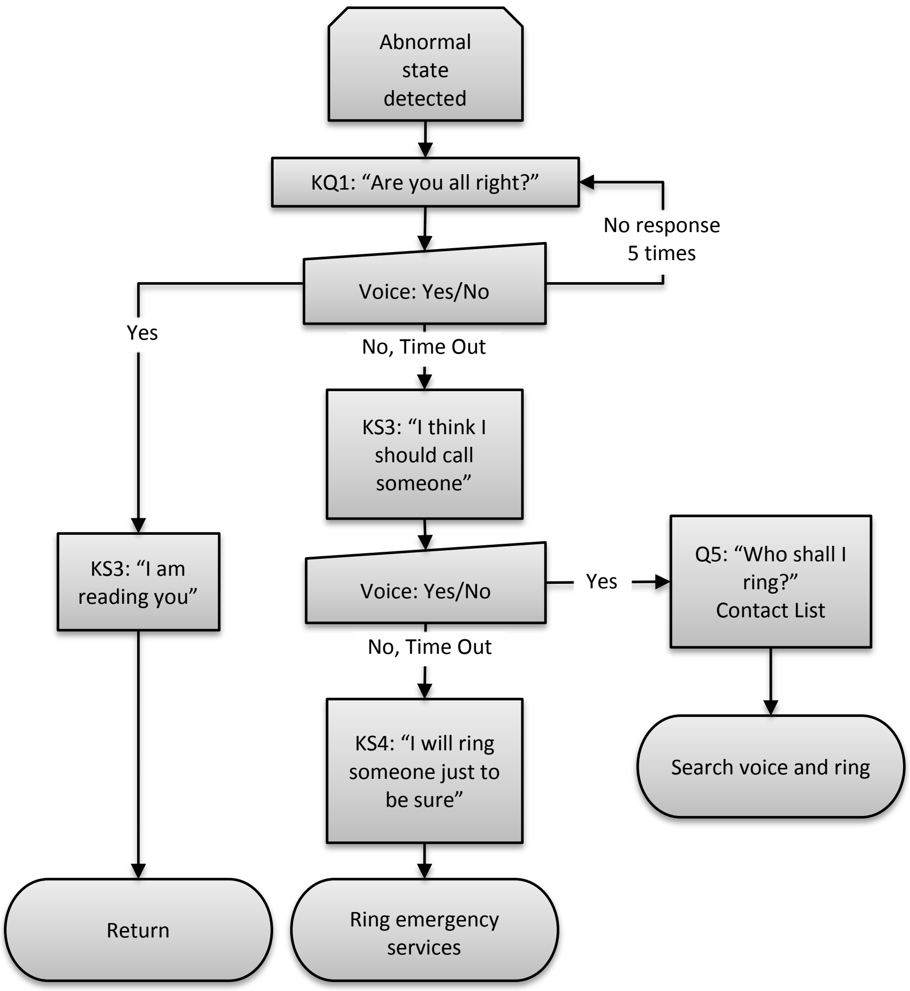


Figure B1 Speech recognition tree used to identify the necessity for calling for assistance in the event of an abnormal state detection

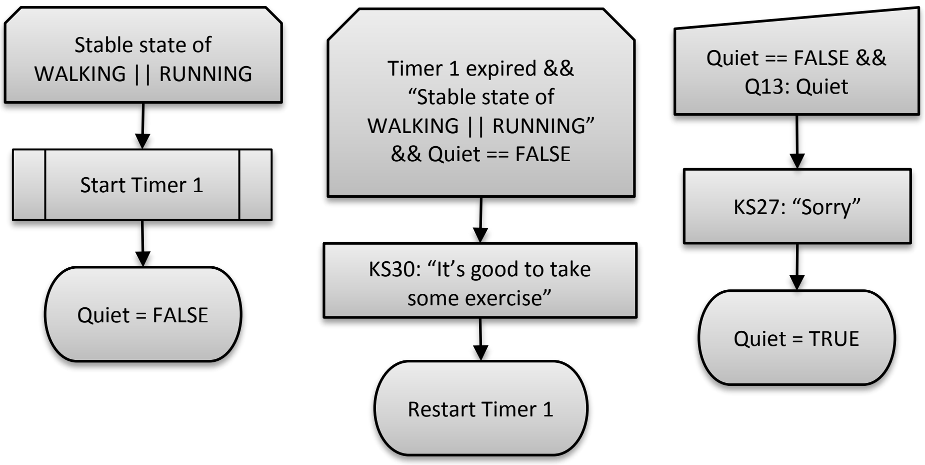

Stable state of WALKING || RUNNING
Start Timer 1
Quiet = FALSE
Timer 1 expired && "Stable state of WALKING || RUNNING" && Quiet == FALSE
KS30: "It's good to take some exercise"
Restart Timer 1
Quiet == FALSE && Q13: Quiet
KS27: "Sorry"
Quiet = TRUE

Figure B2 Speech interaction to confirm the states of motion and silencing of the speech interaction at the user's request

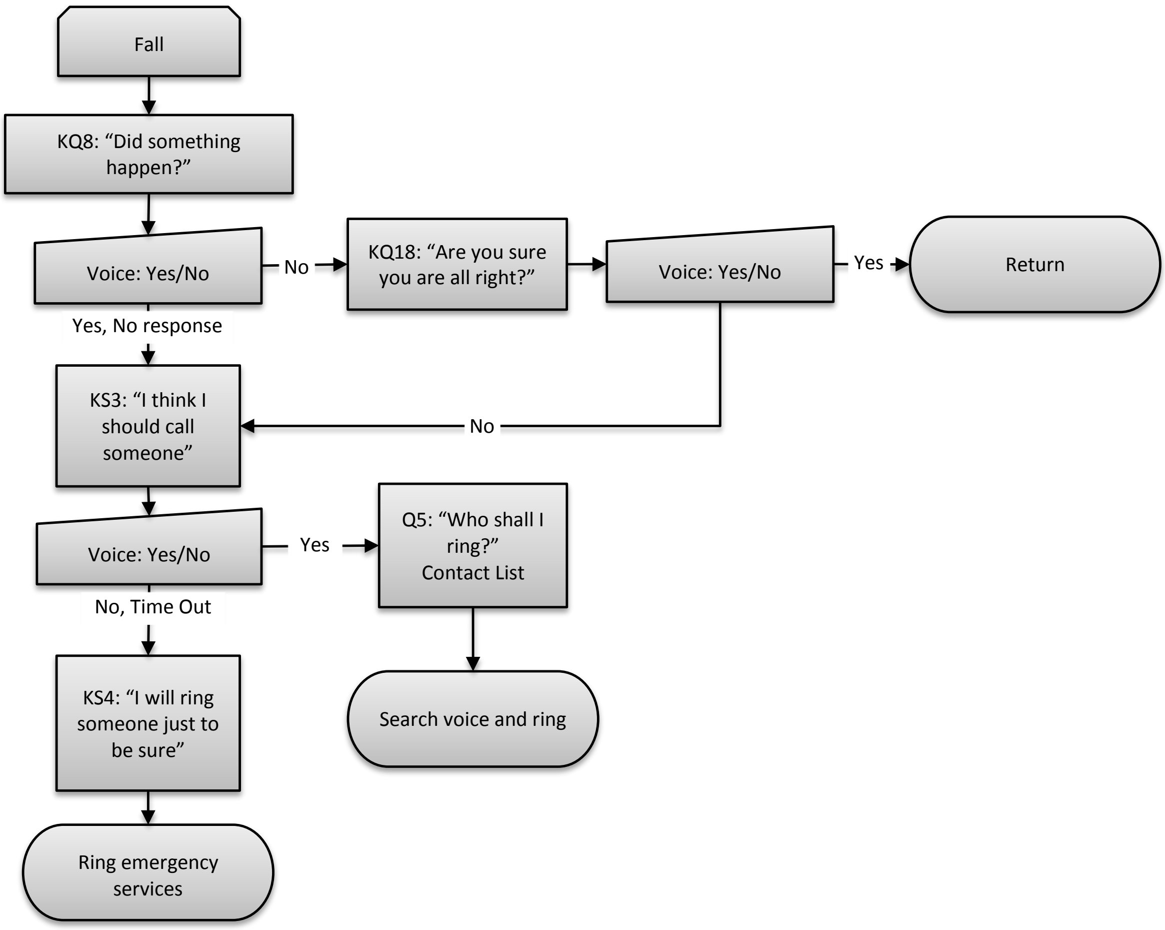

Fall
KQ8: "Did something happen?"
Voice: Yes/No
No
KQ18: "Are you sure you are all right?"
Voice: Yes/No
Yes
Return
Yes, No response
KS3: "I think I should call someone"
No
Voice: Yes/No
Yes
Q5: "Who shall I ring?" Contact List
No, Time Out
Search voice and ring
KS4: "I will ring someone just to be sure"
Ring emergency services

Figure B3 Speech recognition tree used to identify the necessity for calling for assistance in the event of a detected fall

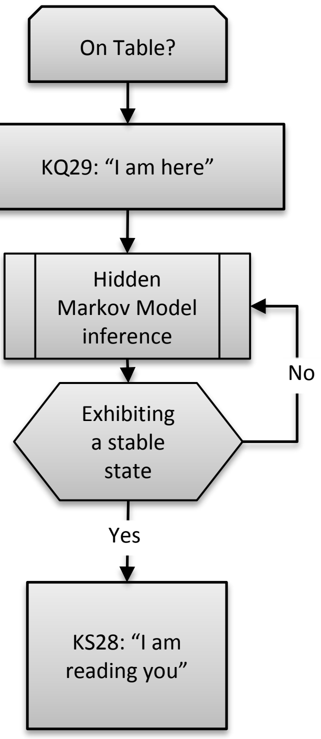


Figure B4 The standard speech interaction tree used to begin the monitoring process

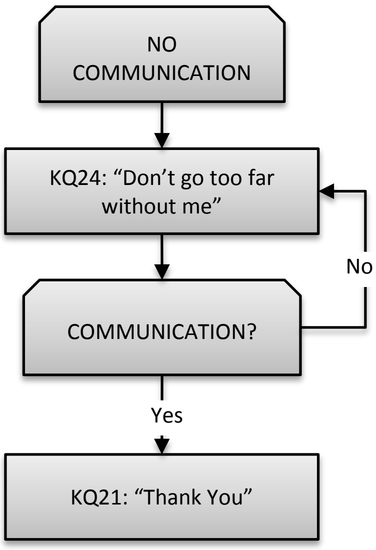
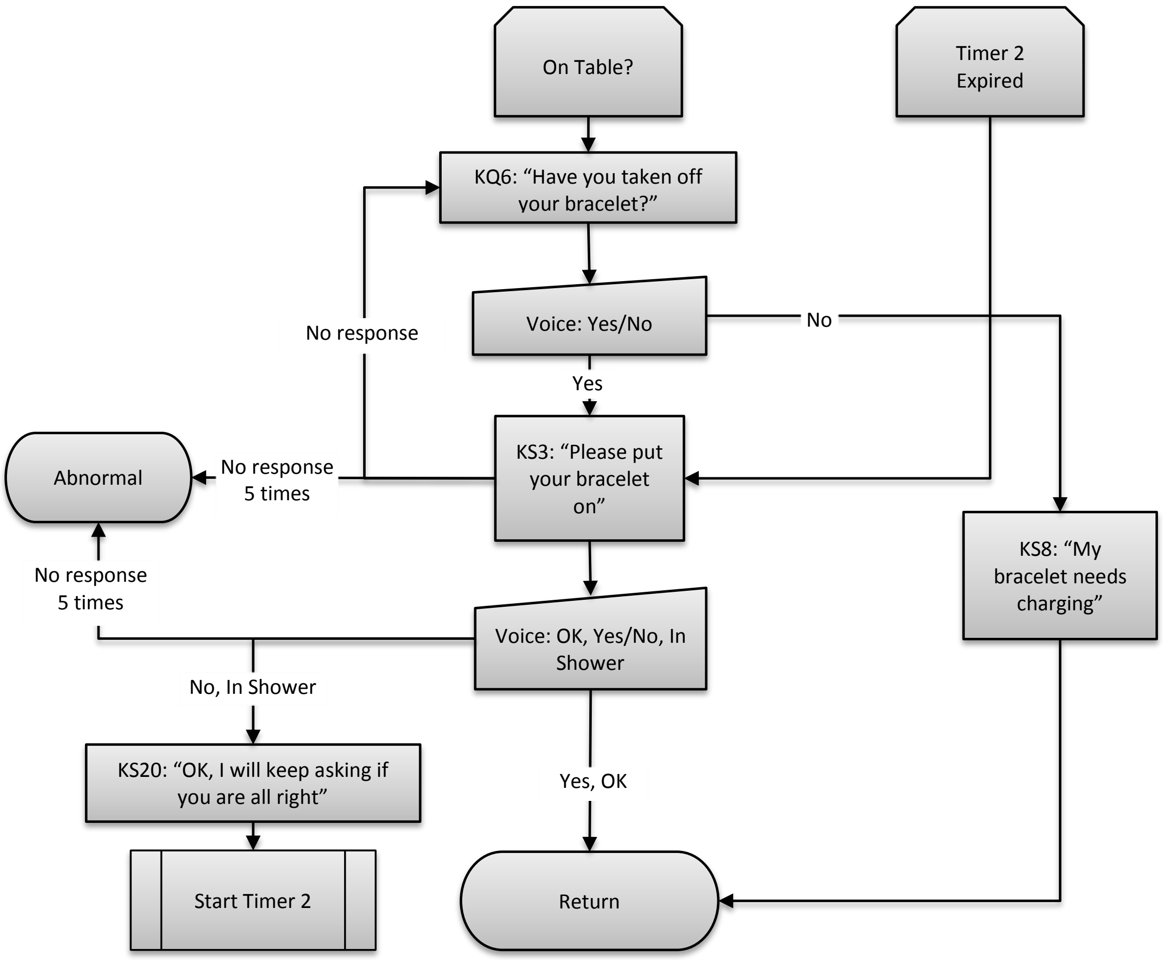


Figure B5 Speech interaction tree to remind the user to take the base station of Verity with them whilst wearing the Wrote



On Table?
Timer 2 Expired
KQ6: "Have you taken off your bracelet?"
Voice: Yes/No
No
No response
Yes
Abnormal
No response 5 times
KS3: "Please put your bracelet on"
KS8: "My bracelet needs charging"
No response 5 times
Voice: OK, Yes/No, In Shower
No, In Shower
KS20: "OK, I will keep asking if you are all right"
Yes, OK
Start Timer 2
Return

Figure B6 Speech recognition tree for an event where the Wrote is removed during the state inference process

Author’s Contribution

J. Winkley, P. Jiang and A. Hossain. “A Behaviour Monitoring System for the Elderly using a Hidden Markov Model and Fuzzy Logic”, presented at RAatE 2010.

J. Winkley, P. Jiang and A. Hossain. “Dimension Reduction for Linear Separation with Curvilinear Distances”. *Soft Computing, Mendel 2011: 17th International Conference on,* pp. 515-522, 2011.

J. Winkley, P. Jiang and W. Jiang. “Verity: an ambient assisted living platform”. *Consumer Electronics, IEEE Transactions on*, vol. 58 no. 2, pp. 364-373, 2012.

J. Winkley and P. Jiang. “Adaptive probability scheme for behaviour monitoring of the elderly using a specialised ambient device”, *International Journal of Machine Learning and Cybernetics*, pp. 1-15, 2012.

J. Winkley and P. Jiang. “Increasing Efficiency in Instance-based Learning with Bloom Memory”. To be submitted to Pattern Recognition Letters, 2013.