Distributed Online Machine Learning for Mobile Care Systems

Submitted in partial fulfilment of the

requirements for the degree of Doctor of Philosophy

at De Montfort University, Faculty of Technology

Ing. Hans J. Prueller, B.Sc.

October 23, 2014

Acknowledgements

I would like to thank my supervisors Dr. Eric Goodyer, Professor Robert John and Dr. Mario Gongora for their support and guidance. I would also like to thank my external supervisor based in Austria, Dr. Ulrich Bodenhofer, for his support and the helpful discussions we had in the early phases of my research. Especially I would like to thank Dr. Jenny Carter for permanently being approachable for discussions, giving inspiring tips and valuable feedback. Thank you for always helping me to keep going and not lose momentum, even in the hardest phases of my studies. Additionally I would like to thank my friends and family for supporting me throughout the course of this research, my fellow students and friends Andreas Huemer and Louise Greenstock, and finally my partner Martina Schrattenecker who acted as an important source of motivation and support.

Abstract

Telecare and especially Mobile Care Systems are getting more and more popular. They have two major benefits: first, they drastically improve the living standards and even health outcomes for patients. In addition, they allow significant cost savings for adult care by reducing the needs for medical staff. A common drawback of current Mobile Care Systems is that they are rather stationary in most cases and firmly installed in patients' houses or flats, which makes them stay very near to or even in their homes. There is also an upcoming second category of Mobile Care Systems which are portable without restricting the moving space of the patients, but with the major drawback that they have either very limited computational abilities and only a rather low classification quality or, which is most frequently, they only have a very short runtime on battery and therefore indirectly restrict the freedom of moving of the patients once again. These drawbacks are inherently caused by the restricted computational resources and mainly the limitations of battery based power supply of mobile computer systems.

This research investigates the application of novel Artificial Intelligence (AI) and Machine Learning (ML) techniques to improve the operation of

Mobile Care Systems. As a result, based on the Evolving Connectionist Systems (ECoS) paradigm, an innovative approach for a highly efficient and self-optimising distributed online machine learning algorithm called MECoS - Moving ECoS - is presented. It balances the conflicting needs of providing a highly responsive complex and distributed online learning classification algorithm by requiring only limited resources in the form of computational power and energy. This approach overcomes the drawbacks of current mobile systems and combines them with the advantages of powerful stationary approaches. The research concludes that the practical application of the presented MECoS algorithm offers substantial improvements to the problems as highlighted within this thesis.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

THIS thesis examines the application of Artificial Intelligence (AI) and Machine Learning (ML) techniques within the context of mobile computing. Because of the light, compact and wireless type of construction of these kinds of systems, their main problems are their very limited computational and energy resources. The scenario of Mobile Care Systems, a special form of Telecare, is taken as a practical use case for this investigation. Telecare systems more generally allow patients to leave the residential care of hospitals. Home Care systems, for instance, remotely monitor the patients in their homes using installed emergency-buttons and sensors. Mobile Care Systems go even a step further and do not only allow patients to leave hospitals and stay in their homes, they even allow the patients to freely move around. These systems are small, portable and battery powered and can be taken with them wherever the patients go - just like a modern mobile phone. A major problem of Mobile Patient Care systems is that they have to reach a long period of autonomous runtime, mainly because most kinds of patients (elderly people, people suffering from dementia, etc.) can not be expected to perform regular maintenance tasks like recharging the batteries.

Modern mobile computing systems (like smartphones for instance) are shipped with powerful processors and/or compensate for their limited resources by shifting complex computational tasks to powerful back-end server systems, which on the other hand produces increased communication efforts. However, powerful processors and data transfer over mobile networks are the most exhausting tasks for the batteries of mobile devices - we all experience this issue by having to plug in the battery recharger of our smartphones and other current mobile devices at least once a day. Because of the requirement of long autonomous free runtime of Mobile Care Systems and their maintenance, it is very important to achieve a very economic and battery-saving operation. This is the point where the inherent difficulty of Mobile Care Systems comes in: To operate as autonomously and economically as possible on one hand, but to be as sensitive and active as possible to never risk the threat of damaging the patients' health at any moment on the other hand. These two main objectives are contradictory and have hampered a wider application and dissemination of mobile patient care systems so far.

This work contributes to Telecare within a practical aspect and to distributed online machine learning within a theoretical aspect, as it presents a novel approach to addressing this difficulty in finding the balance between contradictory aims in mobile care. It balances the conflicting needs of providing real-time complex distributed intelligence, using only limited resources in the form of computational power and energy. The later presented MECoS system approach represents a highly efficient distributed online machine learning algorithm that is capable of adapting itself to the individual patient's physiological situation, by using up as few resources as possible (as it balances computational efforts between simple mobile and complex back-end devices optimally).

This online adaptation allows a most efficient operation of the monitoring algorithm and avoids the generation of costly false alarms. As adaptation itself requires the application of rather resource intensive machine learning algorithms, these learning steps are optimally distributed and balanced between the mobile and a powerful back-end system and at the same time preserve a single knowledge base to maintain high-quality classification results.

Simulation with an implemented prototype compared to classical static client-server systems delivered promising results. The experiments showed that the presented MECoS approach opens new possibilities for the application of mobile patient monitoring systems in healthcare.

This new approach to distributed online machine learning, as argued in this thesis, is meant to be a significant contribution to the field of Artificial Intelligence (AI).

1.1 Goals / Research Aims

Mobile Care Systems have a big problem: their two main objectives are contradictory aims which cannot be achieved at the same time. They ought to

- operate as autonomously and economically as possible,
- be as sensitive and active as possible

Obviously, these aims are leading in opposite directions and additionally strongly depend on the physiological characteristics of the individual monitored patient. This makes it really hard to build applicable and cost-efficient Mobile Care Systems for a broader use. Currently available systems address these problems by focusing on only one of the two aims, being either simple but very autonomous or complex but rather stationary and not autonomous. Both kinds of systems do not really represent an ideal Mobile Care System with all its characteristics, but they bring up a lot of weaknesses, which is the reason why these kinds of systems have not been used widely.

This is the research question that set out this thesis years ago:

"How can we design a Mobile Care System that balances the conflicting needs of providing real-time complex distributed intelligence, using limited resources in terms of computational power and power requirements?"

In other words: The aim of this research project is to discover a novel approach based on Artificial Intelligence (AI) and Machine Learning (ML) methods that results in a high quality and maximum energy- and computationally efficient operating Mobile Care System. This should enable a very economical, long period of autonomous operation without sacrificing the systems classification abilities / intelligence and result in applicable and cost-efficient mobile patient care systems. Broader and increased use of these systems will be the consequence.

Up to now, the two relevant core subject areas of mobile healthcare systems have been treated separately:

1. (AI-based) patient monitoring and
2. mobile computing.

As already mentioned, current systems mainly focus on only one of the two issues and neglect the other. A detailed investigation of these subject areas by taking the possibilities of current distributed systems into account should bring up a smart combination of techniques that will be able to successfully address the research aims of this investigation.

1.2 Structure of the Thesis

This thesis is structured as follows:

- Chapter 2 gives an overview of the state of the art in modern patient care from telecare to mobile care. The specific requirements, problems and limitations of Mobile Care Systems addressed in this research are explained.

- Chapter 3 presents the topic of economic and less-resource intensive data processing or "energy aware computing". This subject area represents a major problem area of Mobile Care Systems besides smart health monitoring algorithms. The basic conditions for artificial intelligence and machine learning on mobile devices are presented and how these approaches can be made more efficient and more economic for the application within the mobile care scenario are investigated.

- Chapter 4 investigates the topic of smart computer-based health monitoring. The issues related to collecting physiological data and computer based classification of sensor data are presented. The difference between static and modern incremental AI-based classifiers is explained. Further, an overview of well-known and popular approaches is given and their possible applicability to addressing the research aims of this thesis is evaluated.

- Chapter 5 highlights the novel MECoS approach based on the outcomes of Chapter 3 and Chapter 4. As presented in the introductory part, a detailed description of the underlying concepts and expectations is provided. An investigation of the key factors for its application, its pros and cons and of alternative approaches is performed. Further the experimental set-up and the results of the simulation test runs are presented and evaluated to test the hypothesis.

- Chapter 6 presents the the hypothesis verification by experiments that have been performed.
- In Chapter 7 the results of the experiments are discussed in context and the limitations of this research are shown.
- The final concluding chapter 8 sums up all aspects of this thesis and gives a prospect onto future research.

Chapter 2

Modern Patient Care Systems and their Problems

2.1 Introduction

THIS chapter presents an overview of known types of telecare systems and explains the requirements, problems and limitations of mobile patient care. The fundamental contradictory aims of Mobile Care Systems inherently based in their requirements are depicted. Finally the research aims of this thesis are derived from the investigated issues.

2.2 From Residential Care to Telecare

Amongst other things, change in working and social behaviours and the progress in medical science and healthcare [141] have led to a higher average life expectancy, compared to 30 years ago: according to the Office for National Statistics (ONS), life expectancy at birth has increased by around 7 years from 1982 to 2012 [141, 142] (see also figures 2.1 and 2.2).

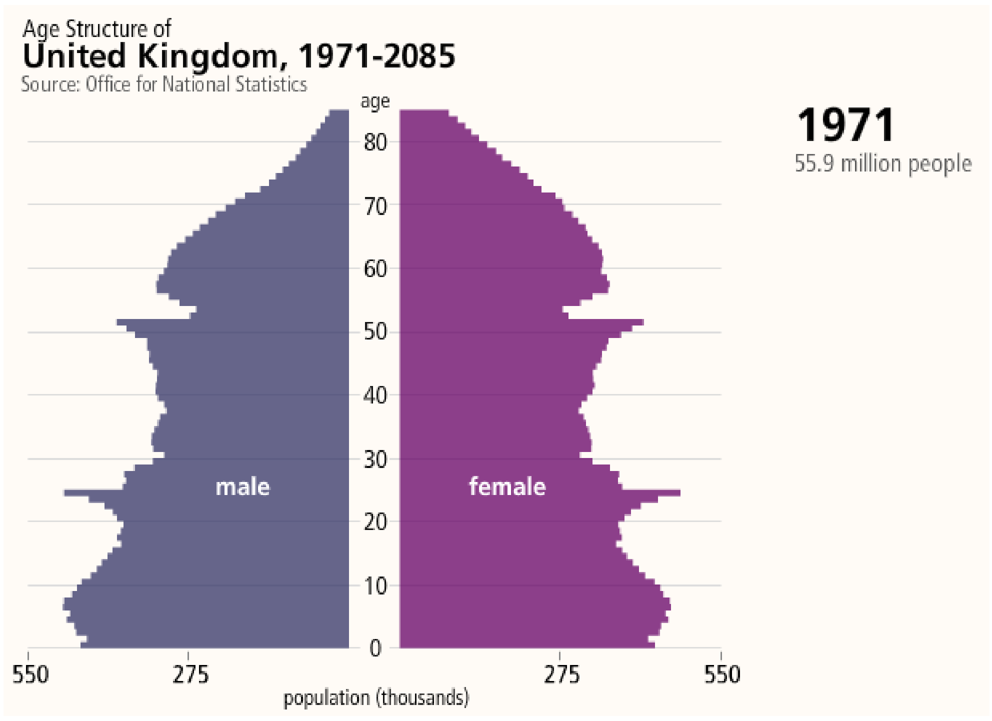


Figure 2.1: Age Structure of the UK 1971 [156]

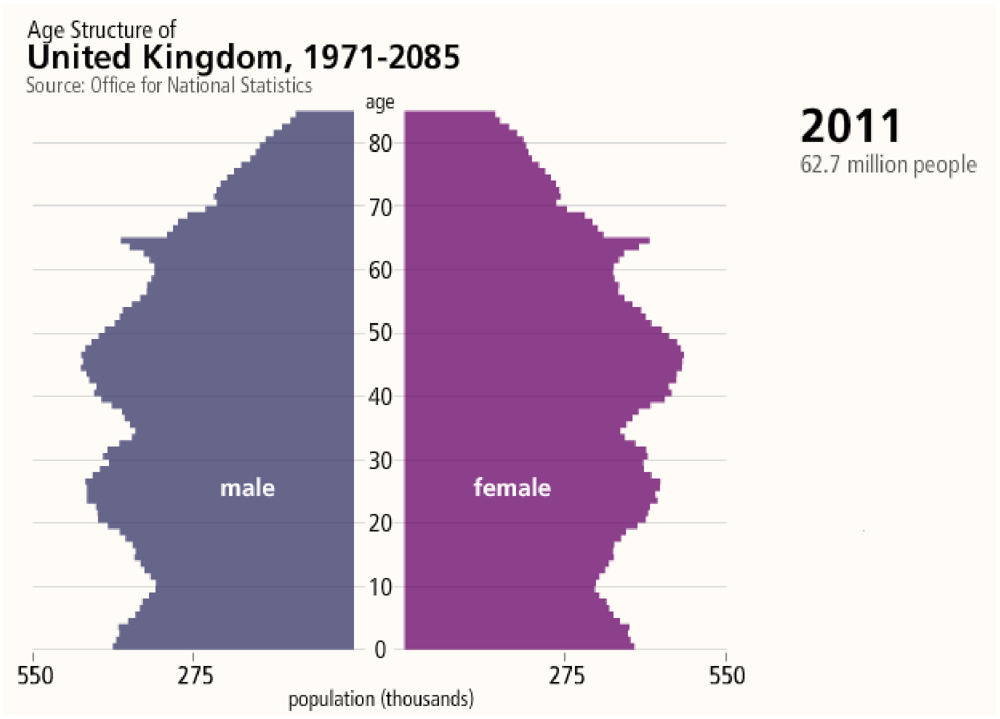


Figure 2.2: Age Structure of the UK 2011 [156]

Many diseases that some decades ago caused massive reduction of the quality of life for patients, can be effectively treated nowadays so that the progression of the disease can be slowed down or even that the patient's living standard reaches a nearly normal level. This progress, however, has its price: the costs for public health systems are increasing permanently [61, 78, 148, 48] and they can hardly be covered by the governments [46]. In addition to several cost-recovery arrangements, for instance additional private insurances and cost sharing etc., the public health systems are forced to perform massive savings due to the permanent cost pressure. Amongst other things it is a major aim to severely reduce the time duration of hospital or nursing home stays, as these represent a very high expense factor[231]. Mainly for patients with chronic diseases and elderly people there is a trend towards non-residential care. As several studies like one of the York Health Economics Consortium in 2009 [28] showed, this does not only provide advantages for the patients who enjoy a significantly higher living standard in their homes, but also massive cost savings for the public health systems that can be achieved through this modern focus on non-residential care [92, 173, 33, 218]. As well as the increased living standard for the patients and cost savings for the public, enabling older people to live independently in their own homes also provides better health results has already been investigated [28, 231, 83, 217, 194].

Further, non-residential healthcare systems are used for prevention and early diagnosis of diseases (e.g. heart attacks), which increases the chances of survival and reduces the risk of permanent disadvantages significantly: About 80% of sudden cardiac deaths happen at home whereas in only 50% of these cases when somebody is watching the incident [36]. Another source mentions a 39% rate of cardiovascular disease related deaths in the UK with 30% of those patients not reaching hospital [133]. Every moment earlier that a patient receives help will dramatically increase his chances of survival and reduce the risk of long-term health-damage. Half of non-watched sudden cardiac deaths happen when patients have been lying for hours before they are found [133]. A permanent non-residential patient care system can help patients to leave residential care and to save lives. There are several types of non-residential care which are applied depending on the type of disease and on economical and financial aspects.

To summarise above paragraphs: Telecare delivers prevention, better health outcomes and increases overall savings to adult care costs.

2.2.1 Residential Care

Residential Care is a kind of alleviated residential care. Residential Care Systems allow the patients to leave residential care in hospitals, but require a permanent stay in so called Residential Care Homes. In these homes, nursing staff and patient monitoring systems are permanently available. This system is effective, if no acute life-threatening disease exists and an ongoing residential stay in a hospital is not required any more, but the need for nursing the patient is still relatively high and nursing staff has to be permanently available (but in a less intensive way than in hospitals).

2.2.2 Tele-Medicine

Tele-Medicine refers to the use of telecommunication technology to deliver health care over a distance ([55] p. 721). Tele-Medicine for instance, can be a very specific type of non-residential intensive care. Intensive Care Systems that are normally only found in Intensive Care Units (ICU) in hospitals are installed at the patients' homes, for instance as investigated by Young et al in [232]. These systems transfer all the data online to a dedicated emergency centre. This type of non-residential care represents one of the most expensive types, as something like a "remote intensive care unit" has to be built.

2.2.3 Non-Residential Care

If the patients are treated in a non-residential way and if the patients are not forced to stay in specific premises as Residential Care Systems do, we talk about Non-Residential Care systems. Non-Residential Care is also called Tele-Health. For instance, Tunstall, a telehealth company based in Yorkshire, UK, consisely defines telehealth as: "Telehealth is the consistent and accurate monitoring of a patient's vital signs and symptoms via easy to use technology in their home." [1].

Telehealth or Non-Residential Care can be split up into Home Care and Mobile Care Systems, which are described section 2.3.

2.3 Non-Residential Care

2.3.1 Home Care

Patients being monitored by a Home Care System, can live independently of residential care in their homes and live their lives at home as normally as possible (e.g. as presented in [87]). Home Care Systems require the installation of specific monitoring methods and devices in the patients' homes (for instance the installation of a communication line with an emergency centre, installation of fall-sensors, etc.), which are permanently watching their health state and set off an alarm in cases of emergency as for instance presented by Costa and Castillo in [45]. The big disadvantage of Home Care Systems is that the safety of the health monitoring will be lost as soon as the patients leave their homes. Therefore the patients are indirectly bound to permanently stay in their homes.

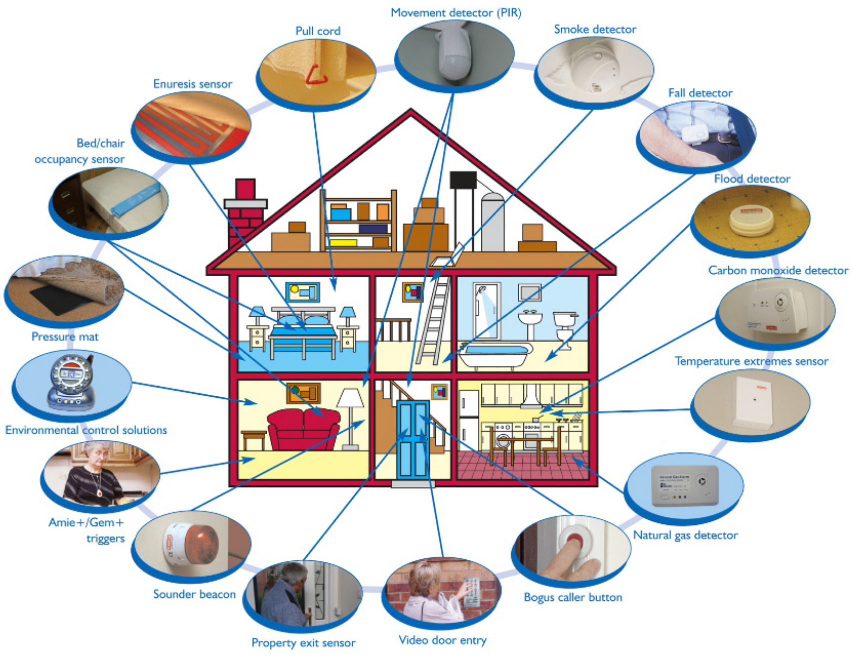


Figure 2.3: Scheme of a Home-Care (Telecare) system as shown in [87]

2.3.2 Mobile Care

Mobile Care is the most modern and ambitious kind of non-residential care. Mobile Care Systems go a step further than Home Care Systems and utilise mobile and portable devices. This enables Mobile Care to ensure the safety of health monitoring of the patients even in situations when they leave their homes. An example for a Mobile Care System is presented, for instance, by Lee et al in [179] called “Cellular phone and bluetooth based blood-pressure and pulse monitor system”. In contrast to Home Care Systems, Mobile Care Systems allow the patients to safely

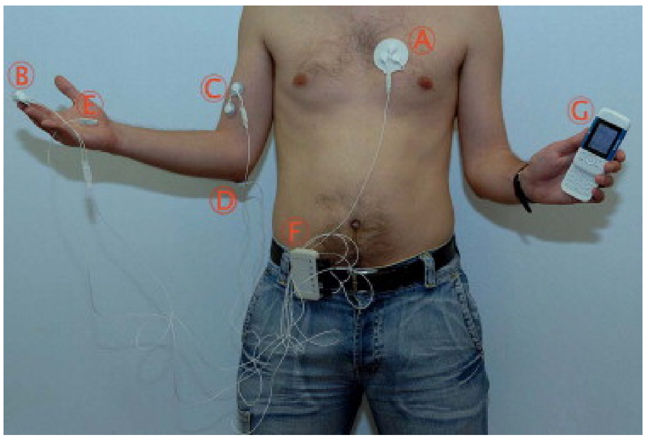


Figure 2.4: Example of a Mobile Care System as described in [180]

leave their homes and perform social activities. Mobile Care Systems provide this advantage by using mobile technologies. This brings about enormous improvements for the living-standard of the patients, but on the other hand also raises several technical problems that have to be solved.

2.4 Problems of Mobile Care Systems

2.4.1 Requirements of Mobile Care

Mobile Care Systems have the aim of permanently monitoring the health state of a patient ideally covering 24 hours a day, regardless of the whereabouts of the monitored person. This should allow the patients to live their everyday lives as normally as possible, leave their houses or flats and participate in social activities and have social interactions. Mobile Care Systems should enable a nearly unrestricted life outside hospitals and independent of residential or semi-residential nursing homes for patients, but provide the security of their health-state being monitored permanently.

Therefore I see the self-explanatory, fundamental requirements of a Mobile Care System as follows:

Safety

Naturally, the safety, i.e. the patient's health is the major issue. The system must not at any point of time risk threats to the health state of the monitored person. The system must not make wrong decisions that may endanger the patient's health ("pessimistic" or over-sensitive way of operation): in uncertain situations, the system MUST set off an alarm in every case, even if it is a false alarm ("false positive"). Every alarm that is set off by the system has to be treated as a possible emergency and medical staff has to examine the situation. A rather simple monitoring of physiological input data which sets off an alarm when specific threshold values are exceeded would not work in this case: either false alarms would be raised permanently (which would make the system in-applicable) or real emergency situations would not be recognised and the life of the patient would be threatened (which would question the system at all). The aim is to transfer the powerful, high-quality and complex Artificial Intelligence based monitoring algorithms from the residential Intensive Care Units to Mobile Care Systems.

Adaptability

It is a very important requirement of Mobile Care Systems to produce as few false alarms (false-positives) as possible but to never ignore a real case of emergency (false-negative). The complex and very powerful AI-based algorithms of the Intensive Care Unit have to be applicable for Mobile Care Systems as well, even though in an adapted, restricted, limited form. Another important requirement within this area is the adaptability of the monitoring algorithms to each individual patient - only this adaptation to each individual can guarantee high quality health state monitoring (as also investigated by Longstaff, Reddy and Estrin[134]). Every human has an individual physiological constitution, therefore physiological data (pulse, heart rate, blood sugar, etc.) also has to be interpreted individually. While the Mobile Care System is working, it has to be able to adapt itself to each individual patient ("online machine learning").

Mobility

Of course, mobility is another self-explanatory requirement for Mobile Care Systems. They ought to be:

- portable, non-stationary:
  the system must not be restricted to a fixed location, it has to be portable and allow the patient to move freely, also outdoors; the system must not depend on a (permanently) wired power supply or network connection.

- easy, wearable:

  the system has to be small, lightweight and as easily wearable as a piece of cloth (e.g. like a belt or a watch) to ensure permanent health-state monitoring with only minimally constricted freedom of movement

Autonomy

This type of health care system has also to be applicable for elderly people and patients with mental diseases, e.g. suffering from dementia. It cannot be assumed or required that patients perform technical maintenance operations or adjustments to the system by themselves - for instance recharging batteries on a daily basis as it is required for modern smartphones. Every required manual intervention of the system, requires medical or technical staff on-site, i.e. visiting the patient where he/she currently resides. This results in major efforts and costs. It is clear that in order to be cost effective the Mobile Care System has to work as autonomously as possible without requiring manual intervention by the patients themselves. The time range of a complete autonomous runtime of the system has to be extended to a maximum possible - a very important point is the duration of the discharge-recharge-cycle of the battery. I.e. reach maximum, completely autonomous work periods of the system.

2.4.2 Limitations of Mobile Systems

Mobile Care Systems have inherent technical restrictions caused by their mobility that are contradictory to their medical aims and requirements. These opposites have to be dealt with.

No Cable Connections (Electricity, Network)

A non-stationary, mobile system must not depend on cable connections. Neither for transfer of data, nor for a permanent cable-based power supply. Nowadays, wireless transfer of data can be treated as non-stationary by using mobile phones and their GSM or UMTS based networks [27, 188, 82, 77, 176]. Many mobile devices, not only phones, have integrated GPRS/UMTS ([27, 77, 188]) modems for wireless data transfer. Another technical achievement as shown in smartphones can be seen in their powerful, high-capacity batteries. They allow a relatively long autonomous runtime of the Mobile Care System without a recharge being required. Nevertheless the requirement of no-cable connections remains one of the major restrictions of a Mobile Care System: how powerful and persevering the mobile resources may be - it has always to be assumed that at some point in time the resource (network, battery) will not be available (any more). Through this assumption the resources should be used in an extremely careful and economic way. It does not really matter how often the in-availability of one of those resources happens, the simple fact that it has to be assumed that it will happen at some time is cause for all the restrictions - the count does not matter (which means: even if the battery power is doubled, tripled or even more, it has to be assumed that the battery will run out of power sometime, so we have to save battery power to achieve the goal of maximum autonomous runtime of the system).

Lightweight, Small

A lightweight and portable system requires a very compact design, in the ideal case the system should be wearable on the wrist like a watch or on the belt like a mobile phone or a pager. The patient should not even notice he's wearing it and it must not restrict his freedom of movement. This emphasises and reinforces the restrictions mentioned above that are inherent with mobile wireless devices - the power supply cannot be guaranteed by simply putting a very large battery into the device, because this would destroy the portability and wearability of the system.

Limited Data

A further restriction of Mobile Care Systems is caused by the fact that mobile peripheral systems have only limited access to data for means of analysis. Because of the compact design and the economic operating mode it is impossible to load or synchronise larger amounts of data directly onto the device, or accessing the data via long-lasting online network connections. Centralised patient monitoring systems - for instance systems within the Intensive Care Unit (ICU) - have the advantage that they have online access to anonymised statistical databases in order to qualify the current situation well. This option does not really exist for mobile patient monitoring systems because of their limited resources.

2.4.3 Summary of the State of the Art

A good overview of state of the art of improved health care by systems consisting of mobile devices and sensors is provided by Darrell West in [219]. Further, Pawar et. al present a framework for the comparison of Mobile Care Systems in [166]. Depending on the specific focus of investigation, there are different ways of performing categorisations of existing systems. In my personal opinion, a very simple but meaningful way is to divide them into 2 classes representing their mobility and their abilities:

- high mobility, but simple (I call them "semi-intelligent" systems)

and

- complex, but with limited mobility (I call them "semi-mobile" systems)

The common problems of all Mobile Care Systems are the conflicting aims they inherently have because mobility always comes with restrictions (limited resources of portable devices). To find the optimal balance between these conflicting aims is very hard, as every patient is an individual with individual characteristics. Existing Mobile Care Systems lay the focus on either one of the conflicting aims and hence, neglect the other.

The first category, wearable and very simple systems, come in most cases in form of only a small, wearable alarm-button or very simple monitoring systems. There are also automatic patient monitoring and detection systems available but they only have a rather narrow focus on very specific events if interest (e.g. automatic fall detection or patient location tracking). In many cases these devices are equipped with a long-lifetime battery and are practically maintenance-free, but they miss complex sensor hardware and sophisticated monitoring algorithms. In many cases they are only equipped with a single physiological sensor, requiring the patient, or other people present, to manually activate the alarm button in case of emergency. Examples of these kinds of systems are:

- Tunstall Telecare and Assisted Living, Tunstall UK, http://www.tunstall.co.uk/ [1] (last accessed May 5th, 2014)
- Medvivo Telecare and Telehealth, Medvivo UK http://www.medvivo.com/ [2] (last accessed May 5th, 2014)
- A simple falling recognition scheme using mobile devices presented by Kim et al in [115]
- FATE - automatic people fall detection service [37]
- Fall detection systems: Principles and approaches as presented by Mubashir, Shao and Seed in [149]
- A location-aware algorithm for dementia care as presented by Vuong et al in [210]

- A cellphone based home care system presented by Figueredo and Dias in [65]
- A Mobile Computing based Fall Detection System as presented by He et al in [80]
- A mobile telephone-based self-care system for asthma control (interactive, not autonomous) by Liu et al as presented in [131]

The second class, complex with limited mobility systems, lay their focus on complex and high quality monitoring of the patients, but take into account the trade-off of limited freedom of movement of the patient. The term "limited mobility" in my opinion also includes systems that basically are mobile systems but provide only limited autonomous runtime, e.g. smartphone based patient monitoring systems. These types of systems consist of several physiological sensors and complex Artificial Intelligence based algorithms to monitor the health state, for instance:

- A Cellular phone and bluetooth based blood-pressure and pulse monitor system by Lee et al [179]
- Mobile Care System for home healthcare presented by Lee et al in [126]
- Body Sensor Network (BSN) or Body Area Network (BAN) based systems as investigated in[132, 133, 180, 121, 119, 40]
- Ozdera et al perform investigation of the smartphone in medicine as presented in [157]

- Lee et al present the development of a mobile phone based e-Health monitoring application in [123]
- Karan et al show diagnosing diabetes using Neural Networks on mobile devices [94]
- The Qualcomm 3G Mobile Health Project [3] http://www.qualcomm.com/media/releases/2011/09/07/ qualcomm-and-life-care-networks-launch-3g-mobile-health-project-help (last accessed May 5th, 2014)
- Istepanian, Philip, Wang and Laxminarayan try to give future prospects on m-health systems as influenced by 4G and emerging mobile systems in [89]
- A mobile system for aerobic activitiy monitoring presented by Reiss et al in [177]
- iCare mobile health monitoring and assisted living system for elderly presented by Lv et al in [136]
- Bodhe, Sawant and Kazi propose a mobile health care system using Android OS and cheap smart phones in [31]

The major drawback of these kind of systems is that they either depend on a fixed installed base station located in the patient's home that has to be in reach, or the battery capacity is simply too weak for the required period of autonomous operation as described above (they require recharging of the batteries at least once a day or more often).

I have found that in most cases the words “battery” or “energy” are not even mentioned in above documents presenting these modern Mobile Care Systems. To me, this indicates the focus of these kinds of systems: they are using modern technologies, state of the art hardware and complex classification algorithms but the objective of maximum autonomous runtime without having to recharge the mobile device on a regular basis is neglected. This is why I call these kinds of systems “semi-mobile”: limited runtime through drained battery power also limits the mobility of the system as a whole.

Annotation: In related work, there is an additional category of systems mentioned as Mobile Care System: Systems that are using mobile phones (or similar) technology to receive physiological signals and send these signals in real-time to other devices e.g. for remote real-time monitoring of patients ([181, 136]). As called by Lee et al in [125], these kinds of Mobile Care Systems are using the mobile devices as “physiological signal extraction devices”. I have dismissed this kind of systems in the categorisation I have performed above, they do not fit in either the first nor the second category. These kinds of systems clearly belong to the group of Tele-Medicine, but from my point of view they are rather some kind of “remote sensor” based patient care than real Mobile Care Systems.

2.5 Summary

To reduce costs of public health and residential care systems and to increase the living standards and health outcomes for patients, non-residential care systems are becoming more and more popular. The most modern and ambitious approach comprise so-called Mobile Care Systems: they have the aim of avoiding any requirements and restrictions for the patients to stay at specific places or remain in specific zones. Mobile Care Systems do not even require the patients to stay in their homes, they aim to allow a nearly normal everyday life. By using modern mobile technologies, the monitoring of the patients' health state is mobile and wearable and accompanies the patients wherever they go. This allows the patients to actively participate in the community and have social contacts. This freedom of movement for the patients is a unique characteristic of Mobile Care Systems, which also bears its inherent problems: limited resources - mainly the dependency on battery power.

Highest Directive: Assure Safety Of The Patient

The highest priority of a Mobile Care System is to guarantee the safety of the patient at any point in time. Simple, static monitoring algorithms are not applicable, because they would produce far too many false alarms (which would make the system obsolete because every false alarm has to be checked by medical staff on-site) or real cases of emergency would not be recognised by the monitoring algorithm and the life of the patient would be at high risk. A monitoring algorithm is required which is able to permanently adapt itself to the individual characteristics of each monitored patient based on its experience and so enhances the quality of classification over time.

Achilles Heel: Power Supply

From my point of view, the power supply is the "Achilles Heel" of Mobile Care Systems. Through development and progress of the mobile phone and embedded computing technology, mobile processors are becoming more and more powerful - the mobile processing and analysis of complex, large datasets does not cause a major problem nowadays (just take the processing power of modern smartphones into account). There are already existing systems taking all these advantages into account, providing complex services as for instance the iCare system prestend by Lv et al [136].
I believe that the real problem though, is the energy consumption and power management of mobile systems (as also mentioned in [178]). Modern smartphones with their powerful processors are equipped with high-capacity batteries, nevertheless, they require recharging daily. This is unacceptable for a Mobile Care System that has to run autonomously over a long period of time (ideally weeks or months).

A very resource-saving, economic mode of operation is a very important requirement for Mobile Care Systems - but this automatically oppones the requirement to never risk the safety of the patient at any point in time. A technical challenge of Mobile Care Systems is to find the optimal balance between those two co-existing and somewhat conflicting aims.

2.6 Conclusion

The technical challenge of Mobile Care Systems is that the two major aims are diametrically opposed:

- to protect the patient's health, a complex, high-qualitative and adaptive monitoring system has to be used (which is very resource-intensive)
- to reach long autonomous runtime, a simple and economic mode of operation has to be applied (which impedes the above)

Because of this antagonism it is impossible to fully address both aims - a Mobile Care System always has to find and assure the optimal balance. Currently available systems lay the focus on either one aim and neglect the other one; there is no system available that tries to address both aims and find this optimal - or at least a better - balance.

2.7 Next Steps

The field of Mobile Care Systems has been described and discussed. Their advantages, weaknesses, the problems and the related issues are shown. The major objectives of this research are generally formulated as well. Next, research work has to be conducted and approaches for possible solutions of the presented issues have to be investigated. The further procedure of this research is identified in the following steps:

1. Find out the state of the art and what others are doing in related fields

   First, the investigation of the field of embedded and energy aware computing is presented as this was my first idea of how the issues of mobile care could be solved. After that, the results of a state of the art review on computer based patient monitoring systems and AI based approaches is presented. With the outcome of both fields of research which are the key components of Mobile Care Systems, a general overview of the topic and addressing the research aims is possible.

2. Construct a solution addressing the research aims and formulate a hypothesis based on the findings and conclusions of literature review.

   A draft has to be made that has the potential to address the previously identified research aims. A hypothesis has to be stated which is the basis on which the solution draft is built upon.

3. Implement a working prototype

   The hypothesis has to be tested by implementing a working prototype of the solution draft which is able to deliver measurable results (that can be compared). An alternative classical or state-of-the art approach that delivers the same measures has to be built, so that the state of the art and the novel approach can be compared with their resulting measurements.

4. Test the hypothesis

   Numerous simulation runs have to be performed and the resulting data is to be collected into a database. Then, benchmarks and comparisons against classical approach measures can be statistically analysed and the hypothesis can be tested.

Chapter 3

Energy Aware Mobile Computing

3.1 Introduction

A very important topic and major research area for the successful implementation of Mobile Care Systems is: resource and energy efficient computing. As mentioned above, health- and patient monitoring systems have their origin in the Intensive Care Unit. There, limited computational resources or even battery powered systems are not an issue at all. Performant classification of alarms and assurance of safety of the patient is the highest directive, for instance alarm specificity and fatigue is a major issue [49, 73, 195, 32, 88]. Hence, current highly sophisticated AI and ML based patient monitoring systems and algorithms for the Intensive Care Units have other issues to deal with and do not really consider these issues [207].

The research subject ”energy aware computing” originally comes from the embedded systems area and investigates economical and energy efficient computing on restricted embedded or portable systems in a broader context (independent from a specific use case or application context). Energy- and resource-consumption characteristics of algorithms are investigated, evaluated and optimised.

The aim of the investigation shown in this chapter is to evaluate issues of energy aware computing topics and find out potential approaches that could be applied in the context of Mobile Care Systems and their issues.

The following sections give an overview of the relevant parts of the much larger field of research in energy aware computing (relevant from my point of view in the context of Mobile Care Systems): general issues for the application of AI and ML based algorithms on restricted hardware are shown, possible ways to enhance and optimise the energy-efficiency of these algorithms are evaluated, further the important part of local versus remote processing is investigated (i.e.: could this be a key-enabler to address the research aims of this thesis?). Finally, a concluding discussion summarises the findings of this investigation and points out the relevant outcomes.

3.2 Local vs. Remote Processing

3.2.1 Introduction

There are ongoing research efforts in energy aware/pervasive computing issues. Some of the researchers are also investigating the question of saving energy by migrating tasks to a powerful remote device. They mainly try to build hardware and application-independent frameworks that should be utilized by mobile device software developers (e.g. software for laptops, PDAs). These frameworks build their decision taking process (local or remote processing) mainly based upon monitoring of execution characteristics, analysis of those data and building up new/adapted rules for future executions (see [66, 182, 185, 184, 120]). This run-time overhead is caused by neglecting the application itself that will make use of the framework.

As the resources are very limited on a remote monitoring device and there is no need to install 3rd party applications, I would suggest another approach in this case: as the monitoring algorithm is chosen at the design stage of the product and there are no dynamic software updates on the monitor device after being deployed/delivered, a pre-production analysis of execution characteristics can be performed- application of analysis frameworks during runtime that try to determine the characteristics of the to-be-optimised software seems not applicable in the context of Mobile Care Systems to me. Certainly the results my proposed pre-production or at system design-stage analysis depend on hardware and its application, but neither the hardware nor the software are the target of major changes after the product has been introduced into the market (i.e. product design changes require new analysis of execution characteristics). Based on this data the local vs. remote processing decision taking parameters can be tuned and fixed installed at the production-series of remote monitoring devices. This results in reduced efforts (processing costs, memory, battery) for the monitor device in production use. In the following sections the important aspects that have to be considered when designing a Mobile Care System are presented.

3.2.2 Basic Rules for Local vs. Remote Processing

Considering the question of local vs. remote processing only from a energy-related point of view, a rule can be derived very easily. Basically, the energy consumed on the restricted (local) device should be minimized. We have to calculate energy required for local (E_{local})and remote (E_{remote}) processing, the minimal value tells us where the processing should take place. Martin et al based their findings on a very simple basic formula in [139]:

$$E_{local} = (P_{systemon} + P_{cardsleep}) * T_{system}$$

$$E_{remote} = (P_{systemon} + P_{cardon}) * T_{trans}$$

where $P_{systemon}$ is the power of the device when processing, $P_{cardsleep}$ is the power of the device when communication is in sleep mode (not transmitting), T_{system} is the time required to complete the processing locally and P_{cardon} is the power of the device when communication is active (transmitting, both directions: sending and receiving), T_{trans} is the time required to complete the job remotely.

Clearly, the above expressions neglect some of the circumstances of the distributed environment, such as network latency, a faster remote device (different local/remote processing times), network conditions, etc. In a subsequent investigation these issues have been integrated into the equations (I will skip writing down each of the single steps here), as a result Martin et al found out that the computation_time divided by dataset_size (Tc/Sd) ratio is a good indicator for deciding between local or remote processing:

A general derived rule says that candidates for remote processing have small transmission size and long computation time, candidates for local processing have long transmission size and short computation time ([139]).

3.2.3 Further Energy Related Issues

Using the basic equations of [139] as a starting point, we can develop our mathematical model (which has a different focus than related research as described above):

$$E_{local} = P_{local} * T_{system}$$

$$E_{remote} = E_{send} + E_{wait} + E_{receive}$$

As sensor data for processing has to be made available in both cases, the energy consumption for gathering sensor data is neglected.

When the algorithm on the monitor node decides to transfer processing to a remote back-end server, the energy consumption equation stated above is not exactly correct: to be accurate, it lacks a constant which tells the energy required to adapt the Artificial Intelligence (AI) after results from the back-end node have been received (i.e. if the monitor has triggered a false alarm, its AI algorithm has to perform a learning step – sometimes this can require lots of processing). More exactly, the equation for remote processing energy consumption would be:

$$E_{remoteComplete} = E_{send} + E_{wait} + E_{receive} + E_{processResults}$$

But as $E_{processResults}$ strongly depends on the AI approach used and is expected to have an overall small fraction of the complete energy consumption, I will neglect it for now.

Power consumption of local processing has to be split up into:

$$P_{local} = P_{procActive} + P_{commIdle}$$

Further, we can assume that the energy required for sending data to a remote server and receiving the results, which are:

$$E_{send} = P_{send} * T_{send}$$

and

$$E_{receive} = P_{receive} * T_{recv}$$

require the same (average) power consumption for tx/rx. We continue with one variable for communication related energy only: $E_{communication} = P_{commActive} * T_{trans}$whereas T_{trans} is $T_{send} + T_{recv}$.

Unlike related research we do not transfer a well-isolated processing task to the remote server (i.e. there is no static data like a file, etc., which has to be synchronized) – in our case – if the system decides to perform remote analysis of the sensor input, data has to be streamed all the time while the connection is open. I.e. we have no clear "send -> process -> receive" flow. In contrast to related research we cannot process the energy required to complete the task (as other projects are investigating well-defined tasks like compiling a programme, formatting a latex document, etc.) – we only can focus on the energy required to process sensor input within a limited duration of time (there is no actual ending of the task "monitor the patient"). So, our formulas will be transformed to:

$$E_{local} = (P_{procActive} + P_{commIdle}) * T_{system}$$

and

$$E_{remote} = (P_{procIdle} + P_{commActive}) * T_{trans} + (P_{procIdle} + P_{commIdle}) * T_{wait}$$

Whereas T_{system} is the time required to process a single input vector locally, T_{trans} is the time required for communication with the back-end system (send and receive) and T_{wait} is the time the monitor device has to wait while the back-end system processes the data. When $T_{trans} = S/r$ as S is the amount of data to be transmitted and r is the transfer rate, we can transform the equation for remote processing to

$$E_{remote} = \frac{(P_{procIdle} + P_{commActive}) * S}{r} + (P_{procIdle} + P_{commIdle}) * T_{wait}$$

Investigations of current hardware (like the wavecomm wismo quik series) have shown, that modern GPRS/UMTS modems ([27, 188, 82, 77]) can be turned off when not used ([13]), so we can assume that power consumption of idle communication is zero:

$$P_{commIdle} := 0$$

so

$$E_{remote} = T_{wait} * P_{procIdle} + \frac{(P_{procIdle} + P_{commActive}) * S}{r}$$

Dynamic Sampling Frequency and Degree of Abnormality:

I would propose a system architecture for building an energy efficient mobile monitoring system to include the technique of "dynamic sampling frequency". Think of the monitor node: its task is to watch an object of interest (i.e. the patient) and identify whether there are exceptional circumstances or not. If not, everything is ok – which means: save energy, work less. If exceptional circumstances are identified (i.e. an emergency, an alarm), the monitor system has to react.

So, the events of interest are the emergencies, not the normal situations. While the monitor is working, most of the time there will be no emergency

– and the monitor has to minimize energy consumption within these periods of time. Dynamic sampling addresses this issue: when there is a normal situation, it is not necessary to read and process sensor data very often ("often" depends on the specific application). Talking about monitoring a person's health, it can be assumed that in the "normal state", reading and processing sensor data at least once or twice a minute should be sufficient (this is just to have an order of magnitude and my personal assumption). Apart from an emergency, where it is necessary to analyse detailed information about the characteristics and gradient of the input data, it may be required to read sensor data up to multiple times a second.

What is needed is a technique to adjust the dynamic sampling frequency (F_s) in relation to the estimation of the classifier, I call it:

"degree of abnormality" D:

As described above, we do not really care if everything is normal. The farther away the situation gets from normal, the more detailed the information about the environment and the circumstances need to be. So if everything is absolutely normal (say: D=0), sampling frequency and local processing is reduced to an absolute minimum. On the other hand, if we have an absolutely abnormal (=emergency) situation (D=1), we need maximum information from the inputs, send alarm data to a back-end system and do not really care about energy consumption. With D we have a tool to influence the dynamic sampling frequency and support the decision taking process of the monitoring AI whether it should raise an alarm or not:

- depending on the value of D, we increase F_s (the sampling frequency)
- if D exceeds a threshold value D_t, the monitor opens up the communication line to the back-end system, raises an alarm and performs remote processing of input data

Introducing the threshold value for remote processing has two advantages: first, we can define a degree of emergency (which is D) where the monitored patient is in a state of danger at which we definitely want to have an alarm triggered, i.e. we have no boolean alarm logic (alarm / no alarm), so we can achieve a "fuzzy" alarm decision by utilizing D as a fuzzy indicator for the alarm.

The second advantage is that the remote node starts receiving sensor input at an earlier stage of the alarm (or even before a true alarm is upcoming). This allows a better analysis of the development of sensor data over a longer period of time on the more powerful (and more intelligent) remote node.

How the sampling frequency is influenced by D depends on the application design and the environment that is being monitored. An example could be:

$$F_s = F_b * (2 - \frac{1}{(1+D)^2})$$

Where F_b is the "base frequency", used in normal state: $D = 0 \to F_s = F_b * 1 = F_b$. Using the equation would increase the frequency up to 1,75 times F_b.

Integration of Data Abstraction and Feature Selection:

Data Abstraction and Feature Selection are methods of reducing processing cost on the restricted mobile monitor device [57, 204]. Clearly, they also blur the input in a certain way. To allow exact analysis of the sensor data in the case of an alarm, feature selection and data abstraction should not be used when data is being streamed to the remote back-end node. It should be incumbent on the remote algorithm itself how the data is pre-processed, the monitor node should transfer the measurements 1:1. Therefore feature selection and data abstraction only affect the energy requirements for local processing.

$$E_{local} = E_{prepare} + E_{process}$$

Here "prepare" means feature selection and data abstraction and "process" means the classification algorithm itself (the AI).

$$E_{local} = T_{prepare} * P_{procActive} + T_{process} * P_{procActive} = (T_{prepare} + T_{process}) * P_{procActive}$$

Now we have to take the sampling frequency into account. Increased sampling frequency produces a greater series of input values for a specific period of time, as the classifier on the monitoring node already receives a single input value representing a time-series (the "feature"), increasing the sampling frequency increases the efforts for data abstraction and feature selection:

$$T_{prepare} = T_{abstraction} + T_{featureSelection} = F_s * c * T_{prepare}$$

Regardless of which technique is more influenced by the sampling frequency (it makes a difference if abstraction is performed before feature selection or the other way round), I have introduced the influencing coefficient "c" which is a constant value that describes the way data abstraction and feature selection are influenced by the sampling frequency.

$$E_{local} = (F_s * c * T_{prepare} + T_{process}) * P_{procActive}$$

Representing Processing Time:

All the above equations strongly depend on time-based values like transfer-time and processing time. The required time to transfer data is already expressed by the throughput, data size divided by bandwidth rate r. As I do not want to rely on a specific hardware platform at this stage of analysis and processing times can only be exact values when being measured on concrete hardware, I'll express the time required for processing as follows:

$$T_{process} = \frac{complexity/size\,of\,the\,algorithm}{performance\,figure\,of\,hardware} = \frac{\sigma}{v_p}$$

Where σ is an abstract value representing the complexity of the algorithm (e.g. number of processor instructions, etc) and $v_p$the performance variable of the hardware platform (e.g. MIPS rate, etc).

Including this representation into our equations for local and remote processing efforts we get:

$$E_{remote} = T_{wait} * P_{procIdle} + \frac{(P_{procIdle} + P_{commActive}) * S}{r}$$

$$T_{wait} = remote\,processing\,time = \frac{\sigma_{remote}}{v_{pRemote}}$$

$$\rightarrow E_{remote} = \frac{(\sigma_{remote} * P_{procIdle})}{v_{pRemote}} + \frac{(P_{procIdle} + P_{commActive}) * S}{r}$$

and

$$E_{local} = \frac{(F_s * c * \sigma_{prepare} + \sigma_{process})}{v_{pLocal}} * P_{procActive}$$

Integration of Data Compression during Remote Processing:

We must not forget to integrate cost of data compression for remote processing equations:

$$Ecompress = Tcompress * P_{procActive} = \frac{\sigma_{compress} * P_{procActive}}{v_{pLocal}}$$

As, similar to data abstraction and feature selection in local processing mode, data compression depends on the amount of input data, we have to take into account the dynamic sampling frequency:

$$E_{compress} = \frac{F_s * d * \sigma_{compress} * P_{procActive}}{v_{pLocal}}$$

$$\rightarrow E_{remote} = E_{compress} + \frac{(\sigma_{remote} * P_{procIdle})}{v_{pRemote}} + \frac{(P_{procIdle} + P_{commActive}) * S}{r}$$

$$E_{remote} = \frac{F_s * d * \sigma_{compress} * P_{procActive}}{v_{pLocal}} + \frac{(\sigma_{remote} * P_{procIdle})}{v_{pRemote}} + \frac{(P_{procIdle} + P_{commActive}) * S}{r}$$

Application of data compression somehow reduces the amount of data to be transferred (S), the compression rate:

$$compressionRate = \frac{S_{compressed}}{S}$$

As the compression rate itself depends on the data being compressed, this is a variable: s (derived of "shrink"):

$$E_{remote} = \frac{F_s * d * \sigma_{compress} * P_{procActive}}{v_{pLocal}} + \frac{(\sigma_{remote} * P_{procIdle})}{v_{pRemote}} + \frac{(P_{procIdle} + P_{commActive}) * s * S}{r}$$

3.2.4 Decision of Mobile Processor

Which component should be integrated on the mobile monitor device? Use a slow but energy efficient or a fast but energy consuming processor? The equations modelled above provide an easy way to decide:

- use a slow but power-saving processor
- or use a power-consuming but very fast processor

on the restricted mobile device. The fast but power-consuming processor consumes less energy than the slow processor if:

$$E_{fast} < E_{slow} =$$
$$\frac{P_{fastProc} * (F_s * c * \sigma_{prepare} + \sigma_{process})}{v_{fastProc}} < \frac{P_{slowProc} * (F_s * c * \sigma_{prepare} + \sigma_{process})}{v_{slowProc}}$$

$$\rightarrow P_{fastProc} * (F_s * c * \sigma_{prepare} + \sigma_{process}) * v_{slowProc} <$$
$$P_{slowProc} * (F_s * c * \sigma_{prepare} + \sigma_{process}) * v_{fastProc}$$

$$\rightarrow P_{fastProc} * v_{slowProc} < P_{slowProc} * v_{fastProc} = \frac{P_{fastProc} * v_{slowProc}}{P_{slowProc}} < v_{fastProc}$$

Then we can define the speed of the fast platform in terms of speed of the slow platform:

$$v_{fastProc} = x * v_{slowProc}$$

which results in

$$\frac{P_{fastProc} * v_{slowProc}}{P_{slowProc}} < x * v_{slowProc}$$
$$\rightarrow x > \frac{P_{fastProc}}{P_{slowProc}}$$

This can be interpreted as follows: *the speed of the fast processor has to be higher than the proportion of its power consumption divided through* *the power consumption of the slow processor to perform better with less energy consumption.*

An Example:

- Slow processor: average power cons of 50mA, performance variable of 5
- Medium processor: average power cons of 85mA, performance variable is 8
- Fast processor: average power cons of 120mA, performance variable is 17

Comparing the slow and the medium processor:

$$x = 8/5 = 1,6$$

power consumption ratio is

$$85/50 = 1,7$$

$$\rightarrow x = 1,6 > \frac{85}{50} = 1,7 \rightarrow 1,6 > 1,7$$

which is false, 1,6 is not greater than 1,7 –> in this case the slow processor would be more energy efficient.

Comparing the slow and the fast processor:

$$x = \frac{17}{5} > \frac{120}{50} = 3,4 > 2,4$$

which is true. In this case, the fast processor would be more energy efficient even if it consumes 2 times more energy than a slow processor.

3.2.5 Rules For Remote Processing

Considering only energy relevant issues will maximise battery runtime but will not result in qualitative classification results and a practical application of the system. Certainly issues other than a maximum reaction time (T_{max}) of the application, specifity and sensivity of the system (similar to requirements in the ICU, e.g. [49, 73, 195, 32, 88]) etc. have to be considered. Thinking of a remote monitoring application and the major aim of this project stated in the introductory part of this document, what are the reasons why the remote device should raise an alarm and delegate control to the back-end node? My initial (and from my point of view self-evident) ideas are:

- if more energy is required for local processing than for remote processing
- if more time is required for local processing than for allowed maximum processing time and remote processing time (time for transmission and waiting)
- and most important: if the "degree of abnormality" is greater than a defined threshold value (i.e. if an alarm is identified / a case of emergency is found)

3.2.6 Communication / Data Transmission

If the remote monitor identifies an abnormal situation, the GSM/GPRS line has to be opened up and sensor data is streamed in real-time to the back-end server to allow detailed and more sophisticated analysis. It has to be considered if data compression of the streamed real time data is applicable - power consumption of the mobile device while streaming the data must be minimized, i.e. as long as the bandwidth capacities are sufficient, no compression should be used to avoid additional energy consumption through calculating the compression algorithm (at least this also produces slight delays of the real time data sent).

Maximum transfer rates for remote devices represent basic conditions for the amount of data being processed at maximum. The available maximum rates strongly depend on the wireless transmission technology that is chosen. The required bandwith on the other hand depends on the specific patient monitoring application, i.e. which physiological sensors are available and what is the dataset width? The more physiological sensors used, the higher the required bandwidth for transmission of the data to the backend will be.

I would suggest to always use the slowest technology available, but one that provides enough bandwidth to be able to transfer the full dataset width to the backend. I think it can be expected that it will also be the most cost-efficient and stable one, e.g. compare ([27, 188, 82, 77, 176]):

- GSM->GPRS

- GSM->EDGE
- UMTS->HSDPA(+)
- new technologies/standards (4G->LTE)

3.3 Increasing Energy Efficiency

3.3.1 Existing/Related Research

There is ongoing research within the field of mobile/pervasive computing which aims at saving energy and balancing the processing of mobile devices. The main targets of these efforts differ slightly from the requirements of a mobile monitoring system: existing research only focuses on energy saving, therefore processor intensive tasks are delivered to a remote server to save battery power (as described in [139] for instance) – in our case this is not possible:

The transfer of processor intensive tasks to an additional device to save battery power for the processing of this data only is applicable, if the transfer of the input data and the results to and back from the remote system requires less energy than the local processing of the data itself. In the case of patient monitoring, the input data is not a single snapshot of a situation that has to be processed, measurements are arriving from sensors continuously, i.e. we have infinite time series data. It would be required to open up a data connection and permanently stream all the input data to the remote device for processing, which would cost more energy than anything else.

Hence, the local vs. remote processing model for Mobile Care Systems is much more complex and cannot be described in a static set of rules. The main objective is to work as efficiently as possible, i.e. the Mobile Care System should use Artificial Intelligence methods to decide when a connection to the remote system has to be opened and should learn from its own incorrect decisions (i.e. reduce false alarms).

3.3.2 Data Compression

Data Compression for Local Processing

Based on outcomes in [204], data can be compressed without loss of effectiveness. This could be a key point for reducing processing costs on the mobile node (if compression is less costly than analysis) and also the amount of data that has to be transferred in case of transfer to a back-end system is reduced. The main question is: are the costs for local data compression lower than the costs saved by compression?

Data Compression for Remote Processing

While the monitoring device is in remote processing mode, all input data has to be transferred as quickly as possible to the back-end device. For time-critical monitoring applications like health-monitoring it would not be feasible to wait until a useful amount of data for compression has been collected, data has to be transferred as soon as new data arrives. The proposed solution to achieve both goals – send gathered input data in real time and perform compressed transmission of this data – I would suggest a simple algorithm (somehow related to "Run Length Encoding", RLE [172, 18, 35]): for every series of input values (a series represents data from a single sensor): transmit all contents of the input data pool when switching to remote node (empty all buffers). After this, only transfer changed values for input series – if the monitored series does not change, do not transmit anything to allow the receiving machine to check whether the connection is working, send a "heartbeat" event at X-time intervals.

This method displays minimal transmission sizes by allowing real-time processing also on the server machine. It is clear that this approach only works when the monitored data does not have any "random jumps". There should not be any "random jumps" when a person's health is being monitored (pulse does not jump from 70 to 10 to 25 to 90 to 20 ....). How the integration of fluctuation limits (i.e. ignore minor changes of input values) affects the overall performance of the system has to be checked.

This method has been successfully applied to the MECoS system prototype as presented later in this document (see 5).

3.3.3 Data Abstraction

Another method to reduce efforts of the mobile classifier and increase energy efficiency is to introduce data abstraction [186, 47, 206, 17]. Russ suggests the application of data abstraction methods to reduce computational costs for analysing/monitoring time-series data. He calls it transform DATA to STATE and STATE to STATUS as presented in [186].

3.3.4 Dynamic Sampling

As the remote health monitor only has to perform checks if everything is "normal" in regular intervals, the sensor sampling frequency should be reduced in idle mode/normal state to minimize power consumption. As the classifier of the monitoring device detects an abnormal situation, the sampling frequency has to be increased to get more detailed information about the state of the monitored object/patient. In the ideal case the sampling frequency is increased depending on the degree of "danger" the classifier identifies. Data compression and data abstraction can be defined and programmed by the system architect / the programmer independently of the classifier used, dynamic sampling frequency models also depend on the abilities of the classifiers. For instance this is not really applicable for boolean classifiers which would just turn sensor data sampling on or off. The sampling frequency has to be adaptive to the degree of alarm the classifier identifies. Dynamic Sampling has already been presented in section3.2.3.

3.3.5 Energy Scavenging

Recent research brings up a very interesting way of tackling the problem of limited energy on mobile/embedded monitoring systems: energy is "scavenged" from the environment of the monitoring device. Lo and Yang mention work on vibration, temperature difference, electromagnetic field, light- and infra-red radiation-based energy generation in [132]. Further research on energy scavenging for instance is presented by Paradiso and Starner [164], Shenck and Paradiso [193], Chang et al [39], Jean-Mistral, Basrour and Chaillout [91] or Andia Vera et al [22].

3.3.6 Remote Processing

Another strategy to increase the energy efficiency of mobile algorithms is to delegate the job to a remote device that can perform the computations with less effort. This only makes sense if the efforts to transfer the input data, to wait for the processing results and to receive the results on the mobile device are below the efforts of local processing. A more detailed discussion is presented in the following chapter.

3.4 Machine Learning on Battery-Powered Systems

3.4.1 Introduction

Due to limited processing resources, a computationally efficient learning and classification algorithm has to be implemented on a restricted mobile device. As stated in [144], machine learning can be seen as the search for the correct target function within a large hypothesis space. In the context of monitoring a person's health there is a huge range of hypotheses, the search with available resources on a mobile device would take too long. It is clear that more sophisticated approaches have to be investigated to solve this problem. Additionally, the restricted energy resources have to be considered [170]. Even though mobile care has to look for a "simple" learning environment on the restricted mobile device, the decisions that have to be taken are essential: if the mobile node falsely decides to have a dangerous situation and opens up the costly GPRS/UMTS line [38, 208, 64, 4], battery power is consumed without reason. If the mobile node decides to wait before triggering an emergency alarm, the patient's health could reach a dangerous condition – in the worst case the patient could die. So: a too pessimistic algorithm will exhaust the energy resources (and reduce applicability of the system), a too optimistic approach threatens the patient's health. Another point that has to be considered is the way common learning algorithms acquire their knowledge: in most cases through exhaustive training experience. Learning algorithms improve according to the increasing number of training datasets they observe. Monitoring a patient's health does not always allow learning from training assumptions, a person's life depends on the system that correctly classifies his/her state of health. The learning algorithm can be "trained" by a set of pre-defined medical rules being representative of a broader population, but these rules cannot be transferred to each patient without being adapted. Every person is different, has a different lifestyle and a different physiological condition. The algorithm has to learn about a person's specific lifestyle and conditions to identify his or her health situation correctly.

So which are the specifics of restricted (i.e. embedded, mobile) machine learning compared to other machine learning systems (standalone, multi-agent systems, etc)? In my opinion the key characteristics can simply be derived by the adjectives “mobile” and “embedded”, which can also be interpreted as: small, robust, portable, battery-powered and with the important need for a long run-time of the battery and minimal need for maintenance.

3.4.2 Power Consumers

Which components/subsystems on the mobile monitoring device consume power and how much? Investigations show that the major power consuming entities on mobile devices are wireless communication/GSM, processing/CPU and graphics/display, as presented for instance by Carroll and Heiser in [38] or Perrucci et al in [170]. Within the context of Mobile Care Systems there are three relevant major power consumers:

- Communication (the GSM/GPRS modem): setting up a connection, tx/rx data, idle or off mode
- Processing (CPU, memory) idle, working
- Sensors (or communication with sensors via Bluetooth, ZigBee) idle, reading data

The main power consumers are Communication and Processing and have to be reduced as much as possible to avoid unnecessary consumption of the battery. Communication with sensors can be seen as being regardless of the used AI based classification algoritm, as all of them have to be fed with the same sensor data (for better classification results, all of them need as much and high qualitative sensor data as available). Of course energy consumption for reading sensor data should be reduced to a required minimum, e.g. by applying the idea of dynamic sensor sampling as described above (see 3.2.3).

3.4.3 Power Consumption Values

Power consumption characteristics of components of the restricted mobile monitoring device of course have an important impact on the overall battery usage and runtime of the system. Modern wireless communication devices offer different power modes, such as "idle" or "off" modes in which power consumption is 0 or only minimal (<10mA)[26, 54]. Some devices also provide a combined, so-called "wireless CPU" instead of a separate CPU and GSM/GPRS modem. Wireless CPU is a GSM/GRPS modem and a powerful CPU in one component ([12, 13, 14, 15]).

For instance, exemplary power consumption numbers of some GSM/GPRS modems show these different power consumption modes and nearly zero power consumption in idle modes:

- Advanced Wireless Planet GPRS Module (tri-band GSM) Power consumption (http://www.gsm-modem.de/gprs-module.html, last accessed on 26.02.2014):
  - ✧ Idle mode: <3.5 mA,
  - ✧ speech mode: 250 mA (average)

- GPS Vehicle Tracking System with GSM Modem Power consumption (http://www.ravirajtech.com/vehicle_tracking_system_india.html, last accessed on 26.02.2014):
  - ✧ GPS Power down, GSM Power ON: 100 mA
  - ✧ GPS Power down, GSM standby: 5mA
- WISMO Quik Q2686 wireless cpu ([15])
  - ✧ power off: 25 microA (minimal power for clock needed)
  - ✧ GSM/GPRS 1800/1900 averages 150-240mA for active communication
  - ✧ 2.3mA in idle mode
- WISMO Quik Q2406, Q2426 wireless cpu ([14, 13])
  - ✧ power off: 5-10 microA (minimal power for clock needed)
  - ✧ GSM/GPRS 1800/1900 averages 150-235mA for active communication
  - ✧ 2-6.5 mA in idle mode

Power consumption of wireless communication with physiological sensors (for instance via Bluetooth or ZigBee [137, 124]) is a constant factor regardless of the software algorithm that is chosen to perform the patient monitoring. Nevertheless, it is part of the global power consumption problem of mobile patient monitoring systems and of course the lower the power consumption that can be achieved, the more the overall system runtime will increase. Similar to GSM/GPRS modems, the short-range wireless communication components consume 0 or only a few micro-Amps in idle mode. When data is transmitted, about 10mA can be assumed [54, 124].

Other important characteristics in power consumption of GSM/GPRS/UMTS modems are the power consumption values in idle and/or active mode, and the fact that all wireless datalinks produce peaks of power consumption when the wireless link is initially opened up and/or data is sent (so-called "bursts" are sent out to the network which is extremely energy consuming). This means that transferring more data over a link that has been opened once is more efficient than transferring less data but with 2 separately required connection instances [26, 54]. As the power consumption values for active data links, idle data links and the opening up of a link strongly vary depending on the specific hardware product, a hardware-independent way to express the efforts for communication has to be found.

Data Transmission Power vs. Data Processing Power

As the above numbers clearly show, data transmission is much more costly than local data processing: GSM modems consume about 200mA on average, embedded processors like the wavecom 24xx family "wireless CPUs" consume avg. 9mA for the processor. We can follow that data transmission is about 20 times more power exhaustive than local data processing – if the execution time is not considered. The question of interest for an application optimising local vs. remote processing would be:

- How long does it take to process the data locally (local power consumption * processing time) and how long would transmission of data and reception of the remotely calculated results take (local gsm modem consumption * communication time)?

This question provides the base for a more detailed analysis of "local vs. remote" processing optimisation. This question is more relevant if there are specific packages of data that have to be processed. In the context of Mobile Care we have ongoing time series data (of the physiological sensors) that have to be processed, i.e. the data "package" is neverending. The question in this specific case is not a classical "should this data package be processed locally or remotely" problem. As data transmission is always more energy exhaustive than local processing on mobile devices ([38, 170, 26]) the question is: how can the number and duration of active data connections to the backend systems be reduced to a required minimum?

As the Mobile Care System has to open the wireless data connection to the backend system in cases of emergency only - and if it is a true case of emergency the draining of energy is not relevant - we have to avoid false positive alarms of the Mobile Care systems (i.e. reduce situations where the mobile classifier makes incorrect decisions). This can only be achieved if the patient monitoring system perfectly fits to each individual patient to better know if a situation is a real case of emergency or not. *Individual adaptation of the Mobile Care System during runtime is a key feature for achieving maximum autonomous runtime.*

3.4.4 Conflicting aims of the restricted device

Why can a simple classifier not be used on the mobile device to reduce energy consumption (e.g. fixed rule sets, if-then-else rules, decision matrix, etc.)?

Apart from the previously mentioned characteristics and already proposed system requirements and architectures – what is the main target of a general monitoring system on a restricted (mobile or embedded) device? The main target of monitoring systems on restricted devices is to reach a maximum autonomous system-run time by providing minimal tolerance against missed alarms. In a nutshell - reach both minimal energy consumption and maximum alarm-hit rate.

Thinking further, maximizing the alarm-hit rate also implies maximizing (or at least increasing) data transmission to a back-end system (as every alarm has to be sent somewhere): Maximizing alarm hit rate means that the system must not miss any alarm, in case of doubt an alarm has to be triggered for safety. In order to avoid missing of a single true alarm, the system has to take increased false alarms into account. This results in increased data transmission which, as a consequence, clearly constructs a conflict: Data transmission is the most energy intensive activity of wireless devices, maximizing alarm hit rate causes increasing energy consumption.

What can be done to attain both goals? Basically, reducing energy consumption of the restricted device can be achieved by reducing the workload: reducing data processing costs and reducing data transmission/communication costs.

Data processing cost can be reduced by screwing down the complexity of the classifier, this seems to be a suitable solution only at first sight: the simpler the classifier is, the more false-alarms are generated. This is caused by the pessimistic approach of the classifier, the algorithm has to trigger an alarm – even if it is unsure: better set-off a false alarm than ignore or miss a true alarm. More false alarms result in more unneeded transmissions, so using a too simple classifier is counterproductive on the other hand. Additionally, the classifier has to adapt its behaviour to the individual patient who is being monitored, so even if the classifier triggers false alarms in the beginning, it has to learn and adapt its future decisions to increase efficiency to a possible maximum. Fixed installed rule sets are not applicable in my opinion. In fact, it seems to be a good solution to use the most intelligent classifier possible: even if this produces increased processing cost, the more intelligent classifier should reduce false alarms by its better classification performance.

To sum up, we can formulate the following rules for the restricted device:

Use the most efficient and most performant classifier (to reduce false alarms as much as possible), which results in fewer unneeded transmissions of false alarms, introduce dynamic sampling frequency of sensor data to minimize energy consumption in idle mode (which works better the better the classifier works) and transfer very processor intensive computational tasks to a remote system if this is more economical than local processing.

3.5 Summary

A very important issue for Mobile Care Systems is energy efficient operation. Following research results originally coming from the embedded systems context, the goal is to deliver an economically and resource saving mobile system - aiming to run with minimal battery consumption.

A major problem derives from the specific context of mobile care: the main objectives of mobile computing systems on the one hand and the main objectives of health-monitoring systems on the other hand are diametrically opposed to each other:

Mobile systems have the objective of working as simply and efficiently as possible - caused by their very compact, light construction and the battery powered system. This means that in the best case, mobile systems run very simple, low computational intensive algorithms which require only minimal (or even no) wireless communication with back-end systems.

Patient or Health-Monitoring systems have the objective of never threatening the health and safety of the monitored person, therefore they work in a very sophisticated, complex and pessimistic manner. This high quality of the classification algorithms and pessimistic mode of operation ensures that no single critical situation is misinterpreted or missed. As a consequence, these kinds of systems require busy communication with an emergency control centre or back-end system on a regular basis and they run very computationally challenging algorithms. This results in health-monitoring systems being inherently complex and power consuming.

These contradictory aims have to be put into an optimal balance - a very difficult challenge for the design of a Mobile Care System.

As the complex AI based algorithms for a high quality patient monitoring system cannot be simply removed or replaced, the energy-efficiency of these algorithms has to be enhanced. There are many ideas to achieve this goal. Firstly I would sugguest by increasing energy of local data processing (e.g. by compression, data abstraction, etc.). A further approach is to optimise the energy consumption by finding the optimal balance between local and remote data processing. Data processing should take place either on the mobile device locally or remotely on a powerful back-end computer system - depending on the least resulting energy consumption on the monitoring device itself.

Throughout this investigation, no generally accepted rules or results have been found which can be applied to all mobile systems. The problem is caused by the high complexity and the many parameters and unknown variables of the global resulting system. For the context of a Mobile Care System I would suggest the general aim can be formulated as:

Use the most efficient and most performant classifier (to reduce false alarms as much as possible), which results in fewer unneeded transmissions of false alarms, introduce dynamic sampling frequency of sensor data to minimize energy consumption in idle mode (which works better the better the classifier works) and transfer very processor intensive computational tasks to a remote system if it works more economically than local processing.

Chapter 4

Computer Based Patient Monitoring

4.1 Introduction

COMPUTER based patient monitoring systems have their origins in the Intensive Care Unit (ICU) (for instance see [49, 73, 195, 32, 88]). Their basic idea is to permanently monitor patients in critical, life-threatening situations 24 hours a day, 7 days a week, automatically, to ensure their safety. These systems should be able to identify critical situations and cases of emergency and alarm medical staff immediately.

Well-proven and enhanced there, these automated monitoring technologies are being applied to more and more other areas, also to Telecare and Mobile Care. The basic principle of all these kinds of systems is the same: physiological and environmental measurements are gathered by a set of sensors. Computer algorithms of different kinds process and evaluate the gathered information. Originally, these algorithms were based on monitoring a small set of parameters and their threshold values and they had only limited classification abilities. Meanwhile there are highly sophisticated and complex Artificial Intelligence (AI) based algorithms which are able to even predict upcoming critical situations before they occur (so-called time series prediction algorithms [217, 168, 163, 235]).

The great advantage of systems within the ICU context is that they are firmly installed, fully controlled systems. They are very cost- and maintenance-intensive - a small number of these systems is centrally provided for a large number of patients [90, 78, 148]. In the context of Mobile Care these conditions are completely different: an independent, personally assigned system is required for each patient. The Mobile Care System has to work wirelessly (i.e. battery powered) and as it allows freedom to the monitored person its operation is hard to determine (see also description of Mobile Care Systems in chapter 2). In order to be affordable and stay economic, Mobile Care Systems have to provide a comparably low "per patient" cost factor (the initial costs plus a very low maintenance effort).

The aim of modern Mobile Care Systems is to apply the proven automated monitoring technologies from the ICU context in an adapted and optimised way for mobile use to achieve the goals depicted above.

The corner pillar for a Mobile Care System is the complex but very economically and low resource consuming monitoring algorithm. Every human being is different and every monitored patient has his/her individual physiological constitution and personal habits. This makes it impossible to feed a mobile patient monitoring system with pre-defined rules or pre-trained algorithms which are the same for all patients. This would lead to a massive number of wrong alarms. As every alarm that is raised by the monitoring system has to be handled, the state of the patient has to be checked by medical staff, which is rather expensive. Hence, too many false alarms not only waste the battery power of the mobile device, they also make a Mobile Care System uneconomical on the one hand and also risk the life of the patient on the other hand, which is not acceptable at all. From my point of view an important feature of every Mobile Care System has to be, that it perfectly adapts itself to each individual patient during its daily operation and optimises its own rule base (based on a common set of pre-defined basic adjustments).

In the following chapter the characteristics of computer based patient monitoring systems are presented and their applicability for the Mobile Care context is investigated. Finally the resulting issues and potential solutions based on the outcome of this investigation are presented.

4.1.1 Common Characteristics of Patient Monitoring Systems

The main focus of a patient monitoring system is to determine whether the patient is situated in a "normal" context or if there is a critical situation, e.g. an emergency. In the best case, the system is able to even predict upcoming cases of emergency even before they actually happen (data classification and time series prediction [217, 168, 163, 235]). Being human, every patient is individual and, depending on physiological parameters like age, weight, gender, shape, etc., has a varied definition of the term “normal” for him or her. To be able to identify critical situations and raise an alarm the health monitor has to process several inputs of physiological and environmental data and process them in relation to the current activity of the patient (also mentioned as “context-awareness” in literature [143, 209, 62, 240]). It should be clear that physiological parameters identifying an emergency while the patient is sitting or sleeping do not automatically identify an emergency while the patient is running or carrying something very heavy upstairs.

Depending on the computational power of the monitoring device and the availability and applicability of sensors, as much input data as possible is required to perform high qualitative monitoring tasks. It strongly depends on the specific case it is used in and the focus of the monitoring system, which and how many sensors are required, for instance the most common are: fall sensors, pulse rate, heart frequency, breath frequency, sweat level, skin temperature, blood sugar and blood oxygen level (e.g. compare presented systems in section 2.4.3). Common to sensor data processing in general, which computer algorithms are most suitable for performing the computational tasks of the system at the highest quality, depends on the application context and focus of the monitoring system.

4.1.2 Gathering Sensor Data

Information has to be gathered by many different sensors which are collecting physiological and/or environmental data. Of course, the sensors have to be attached to or worn by the patient. Hence, it is required that they do not narrow the patient's mobility or only restrict it as minimally as possible. The best case would be that the patient does not even notice the equipment (see also section 4.1.2) [132, 133, 180, 225, 24, 199, 122]. The kind of sensors also strongly depend on the information that has to be collected. One issue of processing sensor data is very common and independent of the application context: noise in sensor data.

Noisy Data

Noise is inherently coupled with processing data that is measured via sensors, it arises because sensors themselves represent a source of error while collecting the data and it has to be assumed that missing or erroneous measurements are read ([237]). As an example, noise arises from not optimally attached body sensors (bad contact, no contact) or negative environmental impacts (distortion signals, magnetic fields, etc.). It is very difficult and nearly impossible to reliably identify wrong measurements or interpolate missing ones. Computer algorithms processing this kind of data simply have to be able to deal with noise.

At first sight it seems that noise in sensor data seems to be a large problem in the context of patient monitoring systems. Noise in sensor data really can be a major problem for some applications relying on exact sensor data, at least if there is only a very low number of sensors that deliver the data and a few measurements that the system relies on. In this case the impact of the noise is very high. Using a higher number of sensors and increasing the number of measurements helps: although it is very hard to completely avoid or filter noise, its impact on the results can be reduced to a level where it is not relevant any more: The impact of noisy data can be reduced by integrating multiple sensor signals at once ([237]). Integration of numerous sensors, abstracting them, reducing impact of noise and forming a single view on the measured situation is an active field of research and also called sensor data fusion. The Mobile Care System PROGNOSIS presented by Pantelopoulos and Bourbakis [162] takes advantage of physiological data fusion. For instance see also [113, 21].

Additionally, in my personal opinion, in the context of Mobile Care it can be assumed that - to some extent - the noise in sensor data is related to the monitored individual and has the characteristic of an additional source of information for the classifying algorithm: a sensor delivers noisy data because of the physiological attributes of the patient, a patient (or his/her nurse) always attaches a sensor in the same sub-optimal way, the patient stays in places with disturbing environmental influences, and so on).

Integration of multiple different sensor signals reduces the proportion of the noise related error within the whole input to the system. As Mobile Care Systems generally focus on monitoring as many physiological sensors as possible anyway, noise in sensor data seems not to be a major issue for

Mobile Care Systems from my point of view. The experiments presented later in this document are performed with an input dataset width of 10, i.e. 10 input values from assumed sensors are used which is already a rather high number.

Feature Selection

The input for the classification algorithm has to represent a trend over a period of time - processing only the last measurement of a sensor would produce an isolated result, being representative for a single moment of time only. Classifying sequential data has to be a view of the development of the values and trends (what was and what will be). The extraction of a value representing a time-series of data is called "feature selection", multiple features for multiple types of inputs are referred to as "feature vectors" ([158, 75, 144, 30, 25]). Feature selection means calculating a value representing a time-series of input values rather than using every single measured input value, for example calculate the average, the median or the mean of a series of input values and use the result as the input for the classifier.

There are various methods for tackling sequential data (how to derive/extract features from a sequence of measurements), always the best-suited approach for the specific case has to be chosen. Examples are:

- sliding window
- recurring sliding window

- Markow models and related (see [57])

Smart Clothing and Integrated Sensors

There is ongoing research work on so called “smart clothes”. Smart clothes are made with special fabrics and equipped with built-in sensors and they are capable of measuring vital values e.g. blood sugar rate. It has to be mentioned that smart clothes worn by humans definitely do not make Mobile Care Systems obsolete, as one might think. In fact the intelligent, smart clothes are acting as a datasource just like sensors for the patient monitoring system as suggested. The only difference is that sensor data is not delivered by - for instance a belt worn device - but rather has to be collected by the monitoring device from built-in sensors and additional external sensors – i.e. smart clothes, watches, belts, etc. The interconnection and communication of those objects clearly opens an additional field of research. “Body area network” is an adequate keyword for this issue but this is clearly out of scope for this research (availability of sensor data for the classifier is treated as given, regardless of the source).

For further information on smart clothing and body area networks see for example: [132, 133, 180, 225, 24, 199, 122]

4.1.3 Classification of Physiological Data

Types of Classification Algorithms

In Artificial Intelligence and Machine Learning, there are two kinds of data classification [144]: One relates to classifying elements drawn from a population, the other one to classifying sequences of data:

- iid = independently, identically drawn (from a population)

versus

- sequential data (time series)

In the case of computer based patient monitoring, it is clearly sequential data classification. Therefore classifiers have to perform so-called “time series analysis” and/or “sequence classification” tasks [57].

Performance Metrics for Classifiers

For an evaluation of different approaches, the definition of performance metrics for the classifiers is required in order to be able to perform comparisons and benchmarks. Based on Zhang ([237]) and Perlich, Provost and Simonoff ([169]) well-established can be:

- sensitivity:

- ✧ [204]: measures the number of correct model-labelled event cases out of the total number of actual event cases
- ✧ [237] measures the percentage of actual alarm-class instances that are correctly classified by the model

- specifity:
  - ✧ [204]: measures the number of correct model-labeled non-event cases out of the total number of actual non-event cases
  - ✧ [237] measures the percentage of actual no-alarm class instances that are correctly classified by the model

- positive predicted value PPV:
  - ✧ [204]: Positive predictive value is calculated by the number of correct model-labelled events, divided by the number of all model-labeled events (correct and incorrect).
  - ✧ [237] measures the percentage of true positives of all the instances that are classified by the model as in the alarm-class. It is one minus the false positive rate.

- accuracy:
  - ✧ [204]: Accuracy is calculated by the number of correct model-labelled sets of derived values (event or non-event) divided by the total number of sets of derived values evaluated
  - ✧ [237] measures the percentage of correctly classified instances by the model over all instances

  - ✧ [169] the number of correct predictions on the test data divided by the number of test data instances.

- receiver operating characteristic ROC:

  - ✧ [204]: “The ROC curve is a plot of sensitivity versus one minus specifcity. Because sensitivity and specifcity can be inversely varied by altering the threshold at which to categorize a case as one class or the other, the area under the ROC curve more effectively describes a models discriminatory ability.”
  - ✧ [169]: “Classification accuracy obviously is not an appropriate evaluation criterion for all classification tasks (Provost, Fawcett, and Kohavi, 1998). For this work we also want to evaluate and compare different methods with respect to their estimates of class probabilities. One alternative to classification accuracy is to use ROC (Receiver Operating Characteristic) analysis (Swets, 1988), which compares visually the classifiers performance across the entire range of probabilities. For a given binary classifier that produces a score indicating likelihood of class membership, its ROC curve depicts all possible trade-offs between true-positive rate (TP) and false-positive rate (FP). Specifically, any classification threshold on the score will classify correctly an expected percentage of truly positive cases as being positive (TP) and will classify incorrectly an expected percentage of negative examples as being positive (FP). The ROC curve plots the observed TP versus FP for all possible classification thresholds. Provost and Fawcett

(Provost and Fawcett, 1997, 2001) describe how precise, objective comparisons can be made with ROC analysis."

- area under ROC curve AUR:
  - ✧ [169]: "Another metric for comparing classifiers across a wide range of conditions is the area under the ROC curve (AUR) (Bradley, 1997); AUR measures the quality of an estimator's classification performance, averaged across all possible probability thresholds. The AUR is equivalent to the Wilcoxon statistics (Hanley and McNeil, 1982), and is also essentially equivalent to the Gini coefficient (Hand, 1997). Therefore, for this work we report the AUR when comparing class probability estimation. It is important to note that AUR judges the relative quality of the entire probability-based ranking. It may be the case that for a particular threshold (e.g., the top 10 cases) a model with a lower AUR in fact is desirable."

Alarm Classes

If the monitoring algorithms identify critical situations, they trigger an alarm. Alarms triggered by the health monitor can be categorised as adapted from [237] and [204]:

- TP-R: true positive, relevant
- TP-I: true positive, irrelevant

- FP: false positive (erroneous)
- or NO alarm (normal) state

The monitoring algorithm itself is not able to decide whether a triggered alarm is clinically relevant or not, neither it is able to distinguish between true positive and false positive alarms. For simplicity, the monitoring algorithm has to identify if the patient is in alarm or no-alarm state. Finer classification and interpretation of the alarms has to be performed by subsequent systems (e.g. at the emergency central the alarm is sent to) and/or human external examiners (for instance medical staff).

Annotation: Another alarm class that represents a special case are “FN: false negative” alarms. A false negative alarm occurs if a certain case of emergency happens that is NOT detected by the health monitor. As the health monitor is not triggering an alarm in this case, FN is not contained in the above list. Certainly, FN is a critical alarm class as its occurrence has to be avoided under all circumstances - a FN could risk a patient life.

Non-Availability of Training Data

Tsien found that the training of classifiers results in more robust algorithms when training data contain more of the events of interest [204]. This means that - in the best case - the classifier is being trained while several real events of emergencies occur. The problem is that this research focuses on other kinds of events than Tsien’s did, in fact it will be difficult to train the classifier with real-world data while the patient is actually having an emergency like a heart attack (...). Tsien’s research focused on the intensive care unit, where events occur on a regular time-basis. Another difference is that Tsien's investigations mainly focused on pre-trained classifiers which are deployed to a production system where no further learning takes place (no incremental algorithm), therefore the quality of the training has much more weight than in this research case: as the Mobile Care System has to adapt itself to the monitored individual "on the job", it is required to apply an incrementally learning classifier which is able to increase its performance over time (the longer it runs the better the results will be). From this point of view it should be acceptable to disregard the quality of training data (at least to some extent) in this case.

Simulation Database: To allow efficient investigation and comparison of different machine learning approaches and classifiers a simulation database containing (virtual) physiological data of patients is needed. For instance the UCI Repository ([5]) provides a set of databases that are used by the machine learning community for the empirical analysis of machine learning algorithms.

4.1.4 Requirements within the Mobile Care Context

There are numerous approaches to artificial intelligence and machine learning, from well-established ones to rather new and experimental kinds. Most of them have been designed for a very specific field of application or case of use. As we need to identify the optimal approach for addressing the problems of Mobile Care Systems, it is necessary to define the most important fundamental requirements to the monitoring system. I define them as:

- it must be applicable for the task of monitoring sequential data and take decisions based on the situation (trigger an alarm or not)
- it must deliver good results although available "training data" is rare (as explained above)
- real-time adaptation (online learning) based on the specific individual and environment is needed ("learn on the job"); after an adaptation phase the system has to perfectly suit “its patient”
- data processing should only require minimal computational resources as it has to work on a limited, portable or embedded device
- minimal use of hardware resources, especially battery (memory, storage, energy) is required

The fact that there are not enough computational resources (memory, persistent storage, processor, battery, ...) on the mobile device generates one major additional requirement which also represents a main research objective of Mobile Care Systems: The system has to work in a distributed way within an unbalanced distributed environment, connected by a wireless (GPRS/UMTS) data line. It has to optimise its distributed decision taking process to compensate for the shortcomings of a restricted, mobile device (and so achieves maximum autonomous runtime without requiring any user interaction, e.g. recharge of the battery).

4.1.5 Individual Adaptation

Physiological classifiers clearly have to take the patient's current "context" into account, physiological activities like sitting, sleeping, walking, eating, running, etc. are important criteria for sensor data interpretation. Additionally, even though physiological measurements in a patient's "idle state" (no activity) will be varying from person to person, the classifier has to learn the "normal" state in relation to the activities (the context) of the patient. This is a clear indicator that "static" classification and monitoring systems will not succeed in the context of mobile care. A relatively young field of research, so called "online learning algorithms" or "incremental learners" (as described later in this chapter, see section 4.2) should help to solve this problem.

4.1.6 AI based Monitoring

There are various kinds of Artificial Intelligence and Machine Learning systems, each one with its individual strengths, limitations and intended fields of application. A presentation of popular approaches can be found in [144, 145, 154, 228, 217, 81, 223]for instance. There are algorithms for speech recognition, image processing, data mining, autonomous vehicle driving systems and many more. It is necessary to identify the ones that can act as a best possible foundation for application within the mobile care context. As presented later in this thesis (see chapter 4), there are mainly two Artificial Intelligence approaches which seem to deliver best performance on sequential data classification: Decision Trees (DTs) and Artificial Neural Networks (ANNs) - and modern derived approaches changing or enhancing these concepts, e.g. [229, 228, 197, 187, 127, 107, 60, 51, 42, 223]. As the unbalanced distributed environment within mobile care provides completely different conditions for the monitoring system [202], whether existing monitoring approaches such as ANNs or DTs concepts should be used or if other approaches for the adaptive monitoring algorithm are required needs to be investigated.

Many different approaches for machine learning algorithms have been developed within the last years. It is to be expected that many of them are not applicable to monitoring a person's health from the start, and many others are. This research sustains recent publications on health-monitoring related research and will refer to the most promising approaches. Some of them are well known in computer science: Decision Trees (DT), Artificial Neural Networks (ANN), Support Vector Machines (SVM), Genetic Algorithms and Genetic Programming (GA, GP), Artificial Immune Systems (AIS), Multi-Objective Systems, Evolving Systems and more. The applicability of the classifiers for monitoring a patient's health state has to be investigated, the classifier has to be able to adapt itself to the individual patient. Monitoring a person's "health" is a very general term for detecting an abnormal physiological state. Which abnormal patterns are of interest, e.g. regarding the risk type a patient belongs to, has to be defined.

With regard to [151] there are mainly 6 fields of problems addressed by intelligent systems:

- Diagnosis

- Selection
- Prediction
- Classification
- Clustering
- Optimisation
- Control

The research within the Mobile Care context addresses the problem of distributed monitoring and decision taking. The monitoring part clearly is a problem of diagnosis while the distributed decision taking part falls in the control (and optimisation) category. Hence, AI approaches addressing diagnosis and control problems seem to be the most suitable candidates for this research.

4.2 Static vs. Incremental Systems

The advantage of incremental or so-called online learning algorithms is that they do not require intensive pre-deployment training as Artificial Neural Networks or Decision-Tree based systems (and all derived hybrid systems) do. They are adaptive and capable of learning "on the job". They are often called "incrementally learning" or "evolving" systems [110, 85, 130, 128, 106, 227, 229, 228, 42, 118]. As already mentioned, in my opinion an online learning system is a fundamental requirement for patient monitoring systems in the Mobile Care context because adequate pre-deployment training is simply impossible (no training data of each individual patient is available).

Online-Learning systems should not be mixed up with so-called “real time” systems: even though online-learning systems are defined as learning and adapting in real time, they may have slight latencies compared to real time systems whose key feature is to provide answers in guaranteed response times - whatever happens. These are highly deterministic systems, required for instance in the intensive care unit (ICU) context: needed for guaranteed real-time reactions. As investigated by [150], in telecare and mobile care where patients are not ICU’ed, so-called “coincidental real time” is sufficient.

Global vs. Local Optimizing Learners

In recent years, many researchers and developers have been using neuro-fuzzy controllers in their projects because they provide working solutions for problems that could not be solved by traditional, "static" (fixed programmed behaviour, non-autonomous) algorithms. Although neuro-fuzzy controllers have become more and more popular for AI tasks recently, several problems came up when being applied to real-world tasks such as the lack of online adaptation, real-time learning, life-long learning and performance problems. Kasabov mentions several investigations in [109, 108] that proved most popular NN (neural network) models like multi-layer perceptrons trained with backpropagation, radial basis functions, self organizing maps, fuzzy neural network models being not suitable for adaptive on-line learning. Most of the state-of-the-art AI approaches are offline learners and only suitable for offline, multi-pass learning. The learning algorithm of those approaches is also called "globally optimized learning". Globally optimizing learners attempt to minimize the output error (this is the difference between the outcome of the system and the expected outcome) by taking all presented training examples into account. This means that the longer these systems are trained the better are the results that are achieved. Processing each single training data set is very time-consuming as the system has to take all presented datasets into account when a new one arrives. Some systems even require multi-pass learning (i.e. multiple iterations over the training data set). A common problem of such systems is "overfitting" (see also [144] for a good explanation of overfitting), i.e. it learns the data used for training and does not generalise well.

The opposite of globally optimizing learners are "locally optimizing learners"[221, 144]. In contrast to the above, locally optimising learners do not try to re-consider all previously seen training datasets when a new one is presented. As the name indicates, they do not try to reduce the learning error globally over all training data sets but they perform a "clustering" of all data sets presented and then optimize the learning error per cluster on a local basis. These algorithms are computationally much more efficient and therefore applicable to online/real-time learning. They also avoid the problem of overfitting [144].

4.3 An overview of well known approaches and their applicability

Many AI based systems exist, each of them with its own targeted field of application. The following sections provide a list of well-known systems that represent fundamental approaches to AI. The list represents the most interesting approaches related to this research in my personal opinion. For most of them, modern enhanced and/or hybrid variants exist. Of course possible fields of application of all of these approaches in a medical context may be found in literature, as it is a very broad field with lots of use-cases and problem domains. The well-known AI based systems are investigated with concise focus on mobile patient monitoring, which is a very specific use case with specific requirements to the monitoring algorithm. Some of the approaches are presented for completeness and not investigated in deeper detail, as their fundamental characteristics show very clearly that they are not applicable within this research context, others that seem to be more promising have been investigated in more detail.

4.3.1 Rule Based Expert Systems

The Rule Based Expert System is one of the first systems to have been developed within the field of artificial intelligence [144]. Its aim is to imitate an expert's "rule of thumb" decisions by processing a large (static) rule base. Once rule based expert systems are equipped with their rules, they are very static and do not have the ability to learn [151]. These kinds of systems is just listed for the sake of completeness, clearly not applicable in the case of a Mobile Care System.

4.3.2 Bayesian Learning

Bayesian Learning, as presented in [144], is a more probabilistic approach to machine learning. It is computationally expensive and requires initial information about probabilities. An example for bayesian learning is the "Naive Bayes Classifier".

- a more complex, probabilistic approach
- computes the most probable hypotheses (MAP, maximum a posteriori)
- has significant computational cost
- requires prior/initial knowledge of many probabilities
- good example in [144], p157

As we do not know anything about initial probabilities in this project and we strictly have to avoid high computational cost, bayesian approaches seem to be inadequate for our research.

4.3.3 Instance Based Learning

Instance based concepts learn by simply storing all data sets presented in a large database. When a new instance arrives, the system reasons by processing all previous instances. An example for instance based learning is the "k-nearest neighbour" algorithm [144].

- does not try to estimate a target function at once,
- classifies instances just when they "arrive" (a lazy approach)
- simply stores presented instances in a large database
- long classification time (nearly all computation takes place when instance arrives)
- therefore inapplicable for real-time learning on a restricted device

As we neither have enough persistent storage to store all datasets on the mobile device, nor enough computational power to perform real-time analysis of all datasets stored in the database, this approach is inadequate for our problem.

4.3.4 Fuzzy Systems

Fuzzy logic transforms computing from the strict boolean logic which we are used to into a more "human-like" way of thinking: computing with words (also called " soft computing"). Fuzzy logic has its strength in dealing with vagueness and impreciseness (for example think of words like "hot","tall","fast", etc...) and therefore is the optimal solution for incorporating human (expert) knowledge into information systems. Fuzzy systems deliver good results in situations where complexity is high and traditional boolean-valued algorithmic solutions are impossible [20],[151].

- handles imprecision, complexity and vagueness
- enables computers to act in a more human-like-way (computing with words)
- fuzzy rules capture (vague) human knowledge -> allows good mapping of inprecise and complex problem domains
- once the rule base is defined, it is static

Fuzzy systems seem to be a promising approach to addressing the complex and inprecise problematic of optimizing the distributed decision taking process. As training data (e.g. medical data of the individual patient) is scarce, pre-defining fuzzy rules - as they are represented by words - is much more eligible than training an Artificial Neural Network or a Decision Tree with only the little amount of training data available [see below]. A major drawback of (pure) fuzzy systems is that the fuzzy rule-base is static which inhibits automatic real-time adaptation of the system - which is a major requirement in our case. Hybrid approaches (neuro-fuzzy systems, evolving fuzzy systems) overcome this problem:

Adaptive Fuzzy Systems

Standard fuzzy logic systems are static. Once the rule base is created it does not adjust itself any more [20]. Literature mainly mentions two approaches to achieving real-time learning / adaptive fuzzy controllers: by combining ANNs and Fuzzy Controllers and/or combining genetic approaches (GA, GP) with fuzzy logic (e.g.[43], [198]). In fact this approach perfectly addresses changing, imprecise and complex real-world tasks: fuzzy logic perfectly deals with the impreciseness and complexity of the problem domain, the neural network bears the learning / adaptation to a changing environment and genetic approaches mainly deal with optimization tasks of non-linear problem domains. As [151] states, the so-called sugeno style inference is widely accepted for adaptive controller problems, in particular for dynamic nonlinear systems.

Pure (static) Fuzzy Systems seem not to be applicable in the context of Mobile Care Sytems to me because of their lack of adaptation to the individual patient during runtime (learning on the job). Adaptive Fuzzy Systems could overcome this problem, they achieve learning capabilities by combining fuzzy logic with (for instance) ANN's or DT's. I would question if a hybrid fuzzy/ANN or fuzzy/DT classifier really is required for a Mobile Care System to achieve the research aims as stated in this thesis. As a first step investigation of classical Machine Learning systems (like ANNs, DTs and similar) without combination with a fuzzy controller seems to be a reasonable approach to me. To test whether the classification results could be even better by integrating a hybrid fuzzy controller with the Machine Learning approach that was selected (ANN, DT or whatever) could be an interesting follow-up research.

4.3.5 Decision Trees (DTs)

Decision Trees represent their knowledge very similarly to rule based expert systems such as IF-THEN rules. In contrast to the static systems mentioned above, decision trees build their rule base by training examples - i.e. they have the ability to learn. Initially, DTs are not incremental learners, i.e. training takes place before deployment into a production environment. In the normal case the rule base does not change any more, once it is running in a production environment. Incremental extensions exist. Examples are the ID3 and C4.5 algorithms [144].

- very popular in many applications
- robust against noisy and incomplete data sets
- good for classification of discrete name-value pairs
- target function has discrete output values
- learns by examining the whole known set of training data statistically, not incrementally
- danger of "overfitting" (emended by "pruning" for instance)
- incremental versions of DT learning have been developed

As the investigation has shown, decision trees perform very well in classifying sequential data for complex monitoring tasks (if they have been trained extensively), also in the medical context as for instance presented in [175, 236, 44, 177]. Although incremental derivates exist, decision trees are not designed for real-time adaptation and require a representative set of training data to achieve good results. The non-incremental types are clearly not applicable to this research project. Incremental approaches could be - computational applicability, real-time behaviour are questions that would have to be answered.

C4.5 Algorithm based Systems

C4.5 algorithm based systems (presented by Quinlan[174]) are based on a decision tree classification algorithm, i.e. basically they inherit their fundamental characteristics as described above. There are also variants of multi-class aware C4.5 based algorithms available, as for instance described in [171]. C4.5 based algorithms are very popular and used in very different fields of application in data classifcation and prediction, e.g. [238, 211, 56, 191].

4.3.6 Artificial Neural Networks (ANNs)

Artificial Neural Networks have been modelled referring to the human brain. They consist of layers of virtual neurons interconnected by "synapses". Knowledge within ANNs is represented by weight values weighting the connections between neurons (the synapses). ANNs learn by adjusting these weight values. Like decision trees, ANNs require intensive training before they can deliver good results in production environments. ANNs enjoy similar popularity to decision trees and are used for many kinds of applications, for instance face recognition, autonomous vehicle driving, etc. ANNs are well suited for nonlinear, complex problem domains without initial knowledge about the problem domain. A drawback is that knowledge within ANNs (the weigth values) have no meaning to human experts and therefore ANNs work rather like a black-box system. The most popular algorithm for ANN learning is the BACKPROPAGATION algorithm [144, 151].

- popular for learning to interpret complex input (sensor) data (but rather to classify it e.g. face recognition than working as a controller)
- robust against noisy input data
- good for real-valued name/value pairs
- requires longer training time than DT
- short evaluation times
- various kinds of learning rules are available, eg:
  - ✧ perceptrons: problematic when instances are not linearly separable
  - ✧ delta rule: "gradient descent" finds a hypothesis with a (local) minimum error ([144] p. 89ff) regardless of linear separability
- multilayer networks represent a rich variety of "decision surfaces"
- recurrent networks apply to time series data; do not generalize reliably
- over-fitting is an important issue
- unsuited to an environment that requires real time adaptation and has only limited sample data [20]

Artificial Neural Networks seem to be one of the most suitable approaches for this research: intelligent monitoring research showed that ANNs deliver very good results in monitoring of sequential data [57, 85, 130, 68]. For me, the fact that evaluation times are short (less computational cost) additionally increases the weight of ANNs in this case. Although this indicates that ANNs are ideal for smart monitoring, they have several drawbacks that are not unproblematic within our unbalanced distributed environment: even though they adapt themselves more quickly than genetic algorithms (regarding to [58]), ANNs require very long training time (computational) and lots of training datasets - both things are scarce in the case of mobile care (it is not realistic to train the monitor with data from a "real" patient extensively, the algorithm has to be trained with common data before and should deliver good results as soon as deployed to the patient). As Ammerlaan and Write state [20], ANNs are not the ideal case for environments with little training data and a requirement for real-time adaptation.

Because they are well-suited for classification and prediction, ANNs are very popular in the context of medical systems [203, 94, 68, 169, 163, 235].

4.3.7 SVMs - Support Vector Machines

Support vector machines learn a concept by trying to maximally separate positive and negative training data instances by a minimum set of data points - the so called "support vectors". These support vectors implicitly define a hyperplane separating the two data sets (positive instances and negative instances); once the SVM is trained, new data sets can be classified just by the stored support vectors [135, 167]. SVMs are used for many classification applications, e.g. classification of text, web pages or image processing [81, 16].

- maps complex problem representations into simpler linear problems

- achieved through "kernel methods" that generate dot-products from input vectors
- "pluggable" kernel functions (problem specific), e.g. radial basis function RBF, linear regression LR, ...
- have proved very effective in real world tasks
- basic SVMs are not incremental although incremental implementations exist

SVMs have a very similar behaviour to ANNs, they deliver good results in classification tasks but require intensive training and do not perform well when real-time adaptation is required and training data is rare.

4.3.8 Genetic Approaches and Multi-Objective Systems

Genetic machine learning is inspired by the concepts of evolution: Species evolve by random mutation and crossings (pairing), due to the environmental impact only the fittest members of a generation survive. The population gets fitter and fitter from generation to generation. There are mainly 2 approaches for genetic learning systems: genetic algorithms (GA) and genetic programming (GP). The more popular (and less complex) GA's fundamentals lie in their approach to encoding a problem scenario into a virtual "chromosome" (i.e. a bit string). An initial population of chromosomes is created randomly, within an iterative process the chromosomes are optimized against the fitness function from generation to generation (pairing/crossing of fit individuals, random mutation). As a result, after several generations the population represents / contains a highly optimized set of chromosomes. The second approach - genetic programming - does not evolve a chromosome but directly generates a programme code to solve a problem following the same fundamental laws of pairing, mutation and a fitness function ([144, 151, 226, 74]).

- a problem scenario has to be encoded into "chromosomes" or programme code directly
- a fitness function determines how well a chromosome or genetic program solves the problem
- "learning" takes place through crossing/pairing of fit individuals and random mutation of chromosomes
- it can take hundreds of runs until an optimal generation of a population is found
- the quality of the result strongly depends on a well defined fitness function, i.e. excellent prior knowledge about the problem scenario is required
- genetic approaches seem not to be really applicable to real-time adaptation

Regarding [151], genetic approaches can be applied to optimisation and data mining tasks very well. As Yao states in [227]: "They do not depend on gradient information and thus are quite suitable for problems where such information is unavailable or very costly to obtain or estimate. They can even deal with problems where no explicit and/or exact objective function is available. These features make them much more robust than many other search algorithms." - this means that genetic approaches also fit well to rather undeterministic problem domains. Nevertheless, from my point of view, a critical issue is the correct encoding of the problem domain into the virtual chromosome and the correct design of the fitness function. Both require profound knowledge of the problem domain - which is missing in the context of mobile patient monitoring case (the specific physiological constitution and living circumstances of each patient differ from individual to individual). Additionally, genetic approaches seem not to be very popular in the online machine learning context to me. Interestingly, they are being used for their well-known strengths in optimisation tasks to train other classifiers or in hybrid approaches ([52, 223]).

In my opinion GA based systems have their focus and strengths on optimisation tasks, where they are very popular and mainly applied. In many cases, optimisation tasks are long running tasks over large data bases (data mining) requiring computational power and time. Genetic Algorithms are also applied to optimise problems with more than one objective, also called Multi-Objective Systems:

Multi-Objective Systems

Multi-Objective Systems have been designed for optimisation tasks which consist of multiple conflicting objectives instead of a single one [138, 239, 63]. As an example, they are used for planning electrical power distribution networks to solve optimisation tasks with multiple variables [187] or optimal supplier selection tasks [192] and further. Multi-Objective systems are based on Genetic Algorithms [187, 63, 239] and therefore inherit the fundamental characteristics of GAs.

They need so-called “objective functions” (or also called test functions) for each aim or objective that has to be solved within the problem domain [153, 192, 239], i.e. the objective has to be clearly identified and excellent knowledge of the problem domain is required to define these objective functions correctly.

As mentioned by Sindhya et. al, evolutionary multi-objective optimization algorithms have shortcomings, such as slow convergence to the Pareto optimal front, no efficient termination criterion, and a lack of a theoretical convergence proof [197].

Like Genetic Algorithms, Multi-Objective Optimisation (MOO) systems and methods are mainly found in literature - as the name indicates - when applied to optimisation tasks. I have presented the contradictory aims of Mobile Care Systems in earlier chapters (see section 2.4). At first sight it may seem that MOO systems could fit perfectly to this research problem as we have opposing aims that have to be tackled. The mentioned opposing aims of Mobile Care Systems are brought about inherently because the fundamental aims of patient monitoring (be as sensitive and pessimistic as possible, i.e. use all resources available to achieve best results) are contradictory to the fundamental aims of mobile systems (save as much energy/battery as possible to achieve a longer autonomous runtime period, therefore use only limited resources of mobile devices and be rather insensitive and optimistic). I have presented the aims of Mobile Care Systems as contradictory or opposing aims, but this is meant more generally and in the context of the system design and the systems basic concept or idea (see also the list of systems that are focusing on either one or the other aspect of these opposing aims in 2.4.3). Talking about contradictory aims, I did not mean that this is the specific task that the Mobile Care classifier algorithm has to solve but rather the application context the system has to fit into. For me, clearly, the task of the AI based algorithm in Mobile Care Systems is classifcation of time-series physiological sensor data by providing best performance in classification of health problems or emergencies. Therefore I do not see Genetic Algorithms or Multi-Objective Systems as classifiers that apply to the task of Mobile Care Systems perfectly.

4.3.9 Artificial Immune Systems

Artificial Immune Systems (AIS) are adaptive systems inspired by biological behaviour like Artificial Neural Networks or Genetic Algorithms. AIS are adaptive systems that mimic immunology and immune functions in a computational context [53]. AIS are applied to complex problem domains like data mining, networking and computer security, optimization, anomaly detection, pattern recognition, clustering and classification tasks [116, 51]. As studies have shown, AIS can result in better classifcation accuracy than Artifical Neural Network based algorithms using the same dataset for classification tasks [60].

As Dasgupta describes in “Artificial Neural Networks and Artificial Immune Systems: Similarities and Differences”, AIS based systems require intensive pre-deployment training and (at least at the time Dasgupta wrote this article), there are no incremental/online variants available [50]. Also Hunt and Cooke describe the learning (“immunization”) of AIS as a pre-deployment task and afterwards deploy the AIS to a real-world task in [86]. Li et al. investigated adaptive AIS and they mentioned that similar to Genetic Algorithms, the knowledge of a so called “affinity function” (similar to the GA’s fitness function) is required that encodes the problem domain:

“Like genetic algorithm, artificial immune systems regard the objective problems which need to be solved as the antigen, and consider the feasible solutions as candidate antibodies. An affinity function is used to evalute the combination of each candidate antibody with the particular antigen. At the end of the evolved process, one or more best antibodies will be found with highest affinity, i.e., the optimal solutions of the objective problem.” [129].

Their experiments showed, that their adaptive AIS optimization “still does not outperform GA in computation time. In the future the goal is to develop a more efficient version of artificial immune systems and to make it meet the requirement of real-time engineering systems” [129].

Dasgupta presents in [50] a comparison of similarities and differences between ANN and AIS. Basically there are many similartities, but how learning takes place is very different. It seems interesting in the Mobile Care context, that AIS are very distributed systems with distributed intelligence. The global behaviour of the system emerges from numerous local interactions. A key technique is the so called self / non-self discrimination or negative selection algorithm of AIS. Dasgupta also mentions, that these kinds of algorithms never reports false positive which would increase the efficiency of health monitoring (no false alarm), but he also mentions that “there may be a chance of false negative” which is an un-acceptable risk in a health-monitoring system (see also 4.1.3). Further he mentions that AIS basically have to learn “self” by collecting “time series (sensor) data that sufficiently exhibhit the normal behaviour of a system”. This indicates to me that AIS - very similar to classical ANN systems - require exact knowledge of the problem domain in terms of intensive pre-deployment training. The physiological constituition and personal life situation of each individual patient in a Mobile Care context can vary greatly. So a key feature of a Mobile Care System is to learn about the normal state of the patient (“self” in terms of AIS) during operation, not during pre-deployment intensive training.

On the other hand, AIS are treated in literature as dynamic and adaptive systems, well suited to permanently changing environments [114, 84]. Hofmeyr and Forrest describe this adaptability in a way that the AIS based system can adapt to changes during operation, but also in their studies the AIS based systems are put into a rather long training phase to learn about “self” before they come into operation. This is a further indication, that exact initial knowledge of the problem domain (in this case “self” or the normal state in terms of patient monitoring) is required and that the AIS systems can adapt to changing environments during operation very well. In the context of mobile care we have a different scenario: the initial problem domain is only known very vaguely and the system has to find out what is the normal state of the monitored person. The environment, i.e. the life and constitution of the patient, is very indeterministic right from the start.

AIS systems are also mentioned to be very applicable when distributed architecture is required ([84, 50]). This is always meant in terms of a permanent network where the components of the AIS are roaming and global intelligence emerges from local interactions of the AIS' cells. In the case of mobile care, a very important aim is to keep the network connection closed as long as possible to save the battery of the mobile device. I.e., the network connection is closed most of the time and the distribution is inhibited. The inherent distributed nature of AIS cannot be shown to be an advantage in this context.

AIS seem to share many characteristics with GA:

- an affinity function determines how well an antibody solves the problem
- "learning" takes place through cloning and mutation [129, 116, 53, 51, 60]
- Artificial Immune Systems can adapt well to changing but initially well-known environments
- Use-cases and fields of application of AIS found in literature are mainly related to computer-security

I would question, if AIS based systems are the best-suited technique for patient monitoring. AIS share many characteristics with Genetic Algorithms which do not seem ideally applicable within the context of patient monitoring to me (as mentioned in the previous section). Generally, AIS are still a very young field of research and probably they provide a viable alternative to established ANN and DT based systems for patient monitoring in general. As they share many characteristics with GA and they do not seem to be really applicable in the field of mobile computing and mobile patient monitoring to me, I will focus on better proved and more established classification algorithms and methods for mobile patient monitoring for now.

4.3.10 Dynamic Neural Networks

There have been efforts to advance the powerful but static Artificial Neural Networks with incremental or so-called evolving features (i.e. build online learning systems based on Neural Networks). Within the last years, there has been significant progress in this field and meanwhile numerous incremental variants and derivatives of ANNs exist, for instance as presented in [42, 118, 111, 103, 106, 96, 213, 224, 128, 29]. Key concepts found in literature are the so-called "Neuro-Evolution" systems presented by Risto Miikkulainen et al ([140, 72, 147, 200, 201, 118, 79, 127]), Xin Yao presented work on "Evolving Artificial Neural Networks" ([227, 230, 233, 228, 229]) and systems based on the "Evolving Connectionist Systems Paradigm" (ECoS) as presented by Nikola Kasabov ([110, 104, 102, 109, 108, 111, 101, 103, 95, 96]). From my point of view the presented approaches are somewhat different and researched independently, but basically they are all about building evolving variants or derivatives of ANN's by dynamically changing the network topology/structure and/or the weights. Xin Yao has presented a good description of the main types of evolution in [227]:

- evolution of network weigths
- evolution of network architecture
- evolution of learning rules
- and mixed/simultaneous forms of the three above

With the rather general term "Dynamic Neural Networks" of this section I try to give focus to my common view about those approaches because the underlying motivation and ideas are quite similar from my perspective. Yao summarises their main characteristic very well in a single sentence: "One distinct feature of EANN's is their adaptability to a dynamic environment" [227] or as Kasabov states a requirement of his ECoS systems, it has to "adapt itself in real time in an on-line mode" ([95]).

In [227], Yao refers to "increasing power of parallel computers" that will make ANN evolution more feasible - i.e. he implicitly states that evolving ANN's in an online way (during runtime) requires high computing power.

As Kohl and Miikkulainen present in [118], Dynamic Neural Network based systems perform well but have problems on high-level decision making problems - they call them "fractured" problems. Like Kasabovs ECoS concept ([95, 96, 110]) , evolving neural network approaches in general have to try to address the "stability-plasticity" trade-off. This leads to a dilemma as presented by Ciarelly et al in [42]: models can obtain new knowledge (plasticity) without forgetting knowledge previously acquired (stability) during the training phase.

What I like best about Kasabovs ECoS idea is that he starts developing his ECoS principle by having a look at well-known problems of current systems and then derives seven major requirements for modern, open neural network based (or as he calls it "connectionist") systems [95]. ECoS itself is only a theoretical construct, with no specific system implementation or technique, described well in [110]. The ECoS paradigm defines seven fundamental laws, an ECoS based system has to obey. ECoS based systems have to

- learn quickly from a large amount of data (fast, single-pass training)
- adapt itself in real time in an on-line mode
- open structure where new features can be introduced at later stages
- accommodate everything that is, has been and will be in an incremental way
- be able to learn and improve through interaction with other COS (connectionist systems)
- adequately represent time and space in different scales
- be able to analyse itself (self-analysis and rule-extraction)

All ECoS systems have to obey these rules but there is no limitation or restriction in how the implementation has to look (e.g. also evolving variants of modern Spiking Neural Networks (SNN) [69] have been developed, for instance as presented by Wysoski and Kasabov in [224] or Kasabov et. al in [106] or Kasabov et. al in [107]). This leaves much room open for the implementation of an ECoS based system. From my point of view the previously shown requirements of Mobile Care Systems and the rules of the ECoS paradigm seem to fit together perfectly. Th ECoS paradigm seems to be a very important foundation for addressing the research aims of this thesis to me.

4.4 Summary

Computer based patient monitoring systems gather their input data from many different sensors. Data measured with sensors inherently contain so-called "noise", which means erroneous or missing measurements caused by external influences that cannot be controlled. To reduce the impact of noise in sensor data, as many measurements as possible (from as many sensors as available) are processed - i.e. the impact of the noise related error is reduced within the whole calculation result. Noise in sensor data is an issue in patient monitoring systems that one has to be aware of, but in the case of modern patient monitoring systems that are classifying time series data with input values from many sensors the impact of noise is minimal. Further, popular approaches for patient monitoring algorithms (as e.g. Decision Tree or Artificial Neural Network based systems) are more robust against noise than others.

Classification of sensor data should take into account the development of values over time, not only single measurements. Single measurements represent a single moment in time and therefore have only very limited significance. Through a technique called “feature selection”, so-called “features” are calculated from a sequence of input measurements which represent the trend of the measurements over time and allow better classification with higher quality results. The calculation of features themselves is a minor problem, but the decision to choose the right calculation method/algorithm for deriving a feature for a specific input value is a very important conceptual task (median, mean, average, ...) and always depends on the concrete application context and specific measured input value. For example, a suitable feature calculation method for blood oxygen level could be calculated in another way than for body temperature or pulse.

Upcoming and brand-new technologies like smart clothing (see 4.1.2) integrate the physiological sensors directly into the clothing of the patients. This development promises more and more available sensors for input data which will allow more complex and better classification of the physiological health state of the patient within near the future.

One of the most important issues when designing an automated health monitoring system is to choose the most applicable and most performant classification algorithm fitting to the specific use-case. The list of available algorithms is rather long, but none of the popular approaches seems to be really ideal for the context of mobile care. A major problem is the non-availability of training data for the classification algorithms, because most of them have to be trained before production use. Otherwise undertrained algorithms deliver results in unacceptable quality. A possible solution could be one of the modern dynamic neural network based online learning algorithms which are able to "learn on the job" and incrementally refine and optimise their classification results over time. On the other hand one of their problems is that they are computationally more intensive and require more computational resources. A trade off between optimal adaptation to each individual patient through incremental algorithms and less resource intensive computing on the mobile device will have to be found, probably by migrating most computationally intensive tasks to a remote backend device.

4.4.1 Further Issues

Health monitoring does not only have to deal with inherently related problems in mobile care and telecare but also in other fields:

- measured physiological data has to be synchronized and combined with "external" (clinical) information, physiological measurements alone do not allow accurate classification of the patient's state (see also [237]) – *this strengthens my approach to integrate a powerful "back-end node" which assists the mobile monitoring device in critical cases* (described later in this thesis). As we have practically infinite storage capacities on the back-end server, we could utilize additional information for classifying a patient's state as soon as the mobile node detects a critical situation.

- simulation database: To allow efficient investigation and comparison of different machine learning approaches and classifiers a simulation database containing (virtual) physiological data of patients is needed. This allows classification and measuement with different computational approaches fed with identical time series data, and comparison of the results.
- the UCI repository [5] provides a set of databases that are used by the machine learning community for the empirical analysis of machine learning algorithms: http://www.ics.uci.edu/~mlearn/MLRepository.html (last accessed June 14th, 2014)

4.4.2 Conclusion

Many well-known and popular AI based approaches for monitoring and classification tasks are available. Comparing their strengths and weaknesses to the requirements of Mobile Care Systems, it seems that there is no single viable solution. In my opinion, there are many approaches that are basically applicable for being used in mobile care. Each one with its specific advantages and drawbacks and it strongly depends on the focus and the objectives of the outcoming system as to which one fits best. For instance:

- fuzzy logic: applies well to uncertain control problems like the local-vs-remote processing problem in this research, a drawback is that pure fuzzy systems are not applicable to real-time learning and adaptation.

- artificial neural networks are the most popular and most efficient learners, but require intensive training with representative data to achieve good results. Combined with a fuzzy logic controller they are a hybrid type of a fuzzy system which is capable of learning, a so-called “neuro-fuzzy” system.
- genetic approaches are ideal for data mining and optimization tasks, but computationally intensive and most applicable for performing analysis of large databases. From my point of view, GA based classifiers should clearly not to be run on a limited portable/mobile device - but they could be used as a complementary part that is carried out on the back-end server. For instance [190] uses GA for self-tuning of a fuzzy logic controller.

In order to find out the best fitting solution for this specific research problem, one must be aware of what is expected from a health monitoring algorithm that has to run on a portable, battery-powered device and should achieve maximum autonomous runtime:

- it must be applicable to the task of monitoring sequential data and make decisions based on the situation (a control problem)
- must deliver good results even if available "training data" is rare (as we cannot know details about every individual patient, training can only be performed based on common but rather general data sets)
- real-time adaptation (learning) based on the specific environment is needed ("learn on the job"), i.e. as the system cannot be trained

extensively, it is required to learn about the individual patient while the system is running in real-time

- data processing should only require minimal computational resources as it has to work on a very limited, portable device (embedded system)
- minimal use of hardware resources, especially battery (memory, storage, energy)

Chapter 5

MECoS - Moving Evolving Connectionist System

5.1 Introduction

PRIOR chapters have presented the field of Mobile Care Systems and their major issues in a general context. Subsequently, the state of the art of the two related main research areas, energy aware computing and computer based monitoring, has been investigated and described.

As the problem area has been thoroughly investigated and analysed, the next step is to synthesise the results and to propose a solution to the highlighted problems and form a hypothesis. In the following chapter, a proposed solution addressing the research aims will be presented.

After a short recapitulation of the key requirements of Mobile Care Systems, the development of the proposed solution, the so called MECoS approach and the ideas behind this approach are explained. Finally, I will describe the way the hypothesis is being tested and the experimental set-up which has been developed to perform this validation.

Requirements

A major problem in mobile care results from the converging of two very different problem areas: health monitoring and mobile computing. The problem is that the inherent objectives of these two problem domains are diametrically opposed to each other. It is simply impossible to actually achieve the aims of both fields at the same time; approaching the aims of one area automatically results in moving away from the aims of the other one. These contradictory aims are shortly revisited as follows:

Aims of health monitoring:

- high quality, computational intensive algorithms
- be sensitive and pessimistic i.e. triggering many alarms - even false ones - is better than missing a single one
- communicate closely with emergency centre/back office

Aims of mobile computing:

- use rather simple, computationally efficient and energy saving algorithms
- be tolerant and optimistic i.e. it's better to save the battery instead of performing unneeded tasks (such as triggering false alarms)

- avoid communicating as it really drains the battery

Current systems treat these contradictory issues at the conceptual stage, but instead of trying to address both aims in a well-balanced way, they emphasise only one group of the aims while neglecting the aims of the other field. This has led to the fact that - in my personal opinion - currently there are two opposing kinds of Mobile Care Systems (see also 2.4.3):

- semi-intelligent systems:
  mobile / wearable, but rather simple without complex abilities (like a simple panic-button wristwatch or a static monitor checking against fixed threshold values - they do allow free movement but the limited functionality makes them not really applicable for real-world mobile care scenarios) [1, 2, 115, 37, 149, 210, 65, 80, 131]
- semi-mobile systems:
  complex, but with limited mobility (sophisticated monitoring systems applicable for mobile care scenarios, smartphone or PDA (Personal Digital Assistant) based - mostly depending on wireless LAN access and only with limited runtime of battery so they do not allow the patients to move freely) [179, 126, 132, 133, 180, 121, 157, 123, 94, 3, 89, 177, 136, 31]

These kinds of systems are suitable for specific and limited use cases only. But in real-world mobile care scenarios, where the system should be able to substitute for medical staff, e.g. monitoring patients suffering from dementia, the behaviour of most patients cannot be predicted and the patients cannot clearly be assigned into one of those two system categories. This reduces the applicability of current kinds of Mobile Care Systems drastically. Most of the patients that would benefit from a Mobile Care System, would require a mixed form of both categories (i.e. a system that brings complex monitoring abilities and allows freedom of movement). From my point of view these "categorised semi-systems" in most cases have more drawbacks than benefits, which clearly inhibits their application and distribution. For instance it is impossible to require a patient suffering from dementia to recharge the batteries of the mobile monitoring device on a regular basis and it also has to be expected that the patients leave their homes (consciously or unconsciously). Nevertheless, a much more complex and resource intensive monitoring system than a simple panic-button will be required (as no one would expect a person suffering from dementia to be able to reliably push a panic button in cases of emergency). Neither a semi-intelligent, nor a semi-mobile Mobile Care System is applicable in this case.

A solution that results in the optimal balance between the opposite aims has to be found: highly qualitative, complex and safe health monitoring while simultaneously providing full mobility and maximum autonomous runtime of the system without required manual interventions on a regular basis (as recharging etc.).

Findings

Literature research covering the two related major topics “computer based patient monitoring” and “embedded/energy aware computing” has been performed. As the review on literature has been completed, a total scenario of the context of mobile patient care systems can be designed. The state of the art of the related major topics “intelligent patient monitoring” and “embedded/energy aware computing” were highlighted and the strengths and weaknesses of several popular methods were shown.

Literature research has delivered many issues and findings. The following steps now have to derive the further procedure from the results of the analysis. Furthermore the rather generally formulated hypothesis has then to be refined into concise and measurable research aims that can finally be tested by doing experiments.

The literature review has shown: there are numerous AI methods and techniques, all of them with specific strengths and weaknesses. Every individual approach is applicable to very specific usage contexts, and simply not or of only limited applicability for others. The general conditions in the mobile care context define high level requirements of the AI-based monitoring algorithms, the strengths of several different AI-approaches are required. Hence, the only valuable solution from my point of view seems to be some kind of a very dynamic system that is capable of addressing the requirements of high-level patient monitoring in a less resource intensive way. A fundamental requirement seems to be the ability of adapting to the individual patient to the highest degree without having pre-deployment training data available, i.e. the algorithm has to be capable of “learning on the job”, i.e. an online learning algorithm will be the starting point.

The investigation of energy aware and embedded computing in chapter 3 has shown that it is nearly impossible to clearly differentiate local and remote processing tasks in the distributed context of Mobile Care Systems. Here the requirement to react very dynamically and optimise this “local vs. remote processing “ question during system runtime exists - another indicator that an online learning algorithm is required.

5.2 My idea for solving these problems

5.2.1 Introduction

As presented in chapter 4, ANN (and ANN-derivates) based monitoring algorithms are suitable for many classification tasks and very popular but they require very specific pre-deployment training and large amounts of representative training data. In the case of a Mobile Care System, the main focus lies on the individual patient in individual life situations which reduces the availability of representative training data to a minimum. Even if there are databases with real, anonymised physiological data for ANN training provided - for instance the “UCI Machine Learning Repository” http://archive.ics.uci.edu/ml/ (last accessed September 18th, 2014)[5] - no specific training data of specific individual patients will be available. The training of the Mobile Care System with the available common training datasets would not result in a system exactly adapted to the physiological conditions and behaviour of the individual patient, which clearly has a negative impact on the overall performance of the system. Further, this would result in neglecting the optimal balance between the opposed health-monitoring versus mobile computing aims and the overall applicability of the system would be suboptimal. I think because of this, traditional, rather static Artificial Neural Networks (and similar classifiers) are simply not applicable within the context of Mobile Care Systems.

Alternative imprecise approaches that do not require intensive pre-deployment training like fuzzy logic based systems would deliver better results than a sub-optimally trained ANN in the beginning, but on the other hand, they do not have the ability to adapt themselves to the patient and his/her individual characteristics during operation (see 4.3.4). I believe that in the long run, for these kinds of systems the quality of the monitoring algorithm will be dissatisfactory and inapplicable in the end.

There are efforts in building hybrid fuzzy systems with other AI approaches, to make up for the missing machine learning abilities of fuzzy systems. In these cases fuzzy systems are combined with ANNs or genetic algorithms to combine the advantages of both kinds of systems (see [20, 23, 43, 76, 93, 155, 190, 205, 222, 234]). From the perspective of Mobile Care Systems, these hybrid approaches also combine the drawbacks of both kinds of systems, so they still require highly specific pre-deployment raining like ANNs or GAs do. They achieve “fuzzyness” in their operation and probably will deliver better results than sub-optimally trained static classifiers, but they still lack the possibility to adapt themselves to the individually monitored patient online during operation (“learning on the job”).

A rather new approach can be found in the field of online machine learning and dynamic Neural Networks (see 4.2 and 4.3.10) . Online machine learning tries to solve exactly the same problems as described above by the advancement of static into incremental algorithms.

The very limited computational resources that are available on a mobile device represent a major issue. Even nowadays, when very powerful mobile processors are available, a long autonomous runtime of the system is more important than the powerful processing itself (= energy efficient, battery saving operation). This is why the system has to be equipped with enery saving instead of superfast-multicore processors and at the same time it prevents the systems from having permanently active communication links with back-end systems. Data communication via mobile networks (GPRS, UMTS and more modern standards...) is the most energy exhausting activity of a mobile device (as shown in Chapter 3).

As described in the presentation of modern patient care systems (see 2.4.3), there are Mobile Patient Care Systems which can operate with very limited resources and achieve a very long runtime on battery power, I call them “semi-intelligent” systems.

In order to increase the autonomous runtime and energy-efficiency of the mobile system, it has to be kept as simple (complexity is more resource intensive) as possible. Anyway, powerful computational resources and complexity is only required in extraordinary situations, i.e. in or shortly before a case of emergency. Most of the time (in normal situations) a plain and simple mobile patient monitoring system would be sufficient.

Why could a rather simple and very efficient mobile patient monitoring algorithm not be combined with a very powerful, stationary back-end system that supports or relieves the mobile algorithm in extraordinary situations? This represents the basic idea of the so-called MECoS approach:

A Mobile Care System that is kept working as simply and efficiently as possible being complemented by an assisting, powerful and more complex back-end system. In combination, these two parts should be able to find the optimal balance between both opposed major aims of Mobile Care Systems. In certain cases, the powerful back-end system takes control and benefits from its more powerful processing capabilities, big historical databases and unlimited power supply (because it is not battery powered). The situation can be classified faster, more exactly and with maximum classification quality.

The Mobile Care System works in a rather simple and pessimistic way. In cases of doubt, it should rather trigger a false alarm than not show any reaction and delegate control to the back-end system for further classification. The back-end system can override the decision of the mobile system and avoid false positive alarms. The remaining problem is that false alarms waste too much of the resources on the mobile device.

To avoid unnecessary false alarms being triggered by the mobile system itself, the acquired knowledge (i.e. the fact that the back-end system classified the alarm as false positive) has to be transferred back to the mobile system. The mobile system itself can stay simple, in cases of emergency a much more complex analysis and classification and a learning algorithm assists on a powerful back-end system. *As a result the newly acquired knowledge is given back to the mobile system i.e. the mobile system learns remotely and energy exhausting false positive alarms during the runtime of the system are reduced to a minimum.* This fundamental concept of a single knowledge base which allows remote learning on local knowledge is shown in figure 5.1.

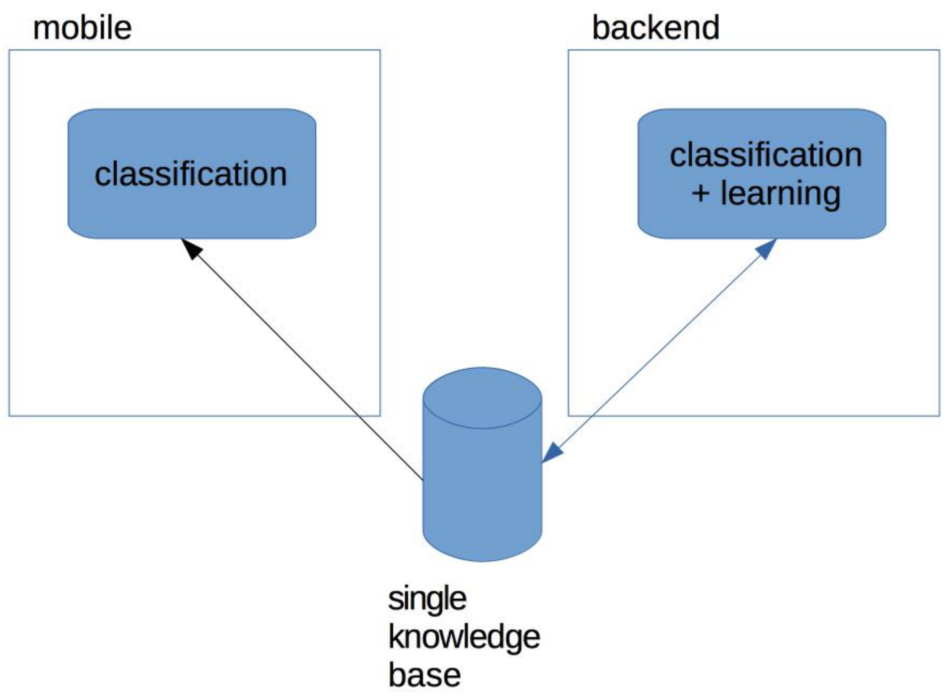


Figure 5.1: Performing resource intensive tasks on a single knowledge base at the backend

The problems arising from the above concept are obvious:

- every false positive alarm that is triggered by the mobile system generates unnecessary drain of battery power, which means the false alarms set-off by the mobile system have to be reduced (at least after an acceptable initial deployment phase). As mentioned above, the mobile system also has to learn from the decisions taken by the back-end system to avoid triggering a false positive alarm in similar situations again. It has to be investigated how this transfer of knowledge or learning on a single knowledge base can be achieved.

- the mobile system "owns" the permanent knowledge base about the monitored patient, the back-end system is only connected under special circumstances for a rather short period of time, i.e. it does not know the history of the patient (the time series data is missing). The sensor data presented between two "connects" is not available to the back-end system. It is utterly impossible to store and transfer data to the back-end system that has been recorded in the meantime (amount of data, duration of transfer, ...) - i.e. in a way that has to be defined later, the mobile system has to be able to transfer its knowledge or its state about the patient to the back-end system.

If above points are considered, it is clear that they are very similar. It is about transferring the learning or the acquired knowledge ("the mind") within a distributed system between the participant nodes and provide distributed common access to a shared single knowledge base.

There are various approaches to distributed learning, for instance so-called multiagent systems [146, 160, 220]. In most cases the nodes in distributed systems learn and act on their own, the global intelligence of the system simply results from the collective actions of the single nodes (compare also: “swarm intelligence” [165, 59, 112]). A specific, central “shared mind” does not exist.

The idea of the MECoS system inverts this approach:

The aim clearly is not to build a distributed learning multi-agent system, much more it is to build a distributed, roaming intelligence: The mind of the monitoring system is not stationary and moves from the mobile device to the back-end device, i.e. the whole learned knowledge is transferred from one node to another. The overall acquired knowledge is available to every node that is currently active, there is only one single knowledge base (a single classifier instance) that moves from one device to another (i.e. from mobile to the back-end).

Having a single classifier with a single knowledge base moving from one node to another eliminates the need to define cooperation and communication rules and allows a very flexible and dynamic distributed decision taking process. The resulting dynamics allow the Mobile Care System to perfectly accommodate itself to each individual patient. Instead of having multiple agents which learn on their own knowledge bases, a moving classifier should increase the overall efficiency of the system by learning on a single knowledge base.

In the standard case the mobile device keeps monitoring the patient as long as the status is classified as normal. In case of a potential emergency, the wireless connection is opened up and then it is *not* an independent server side algorithm that takes control, but the mobile nodes mind moves from the restricted mobile device to the powerful back-end device and resumes working in a different, just much more powerful environment. This step is essential: when moving from the mobile to the back-end device, the knowledge of the classifier has to be transferred into the device. As soon as the classifier is respawned on the back-end device it continues to work as had it been running on this hardware all the time. In addition, the classifier can now use all additionally available resources (i.e. additional input parameters for decision taking) which would not have been available on the mobile device and so can take much more precise decisions.

The following graphic shows a visual concept of the MECoS System approach as described above:

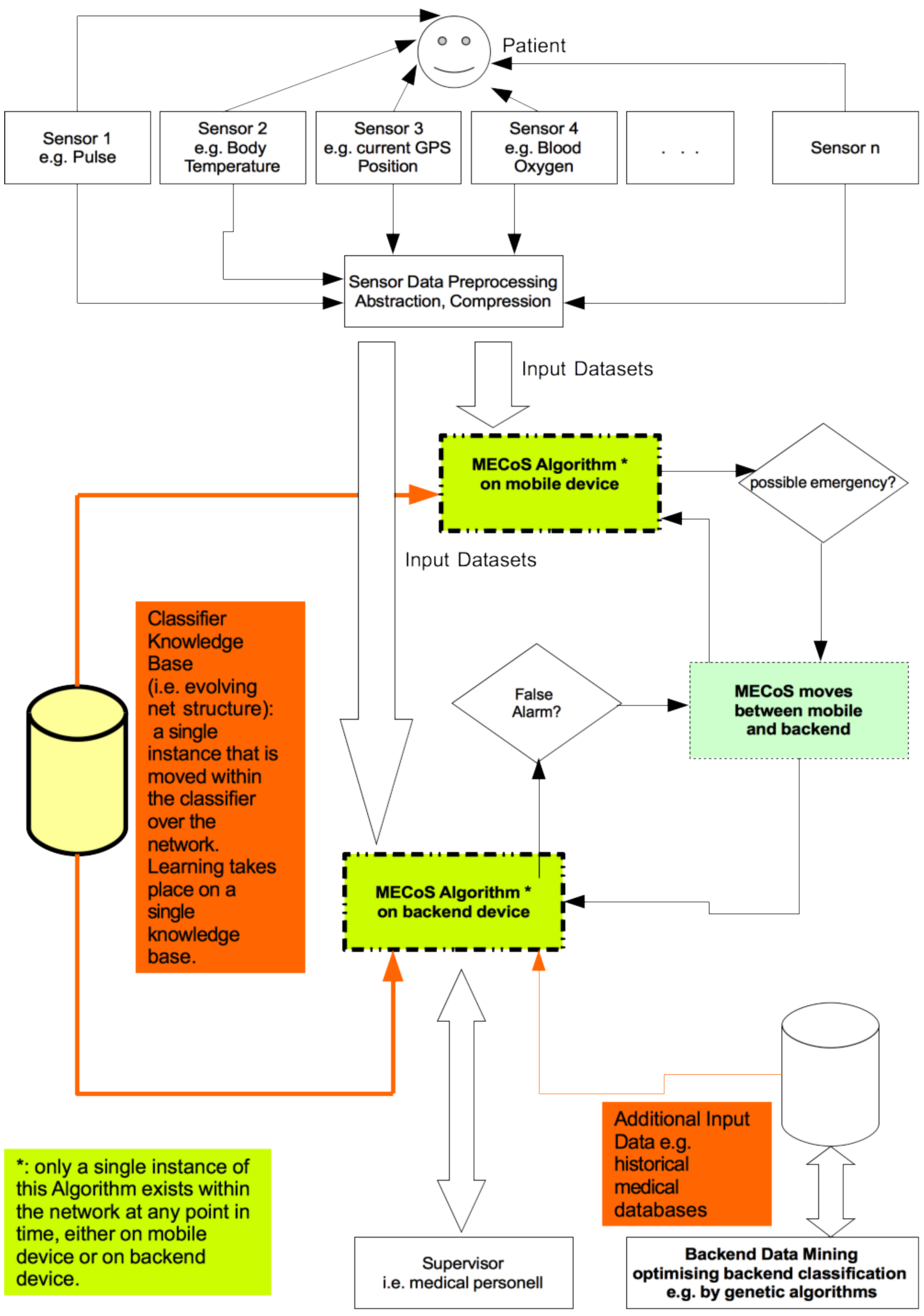

Patient
Sensor 1
e.g. Pulse
Sensor 2
e.g. Body
Temperature
Sensor 3
e.g. current GPS
Position
Sensor 4
e.g. Blood
Oxygen
. . .
Sensor n
Sensor Data Preprocessing
Abstraction, Compression
Input Datasets
MECoS Algorithm *
on mobile device
possible emergency?
Input Datasets
Classifier Knowledge Base (i.e. evolving net structure): a single instance that is moved within the classifier over the network. Learning takes place on a single knowledge base.
False Alarm?
MECoS moves between mobile and backend
MECoS Algorithm *
on backend device
Additional Input Data e.g. historical medical databases
*: only a single instance of this Algorithm exists within the network at any point in time, either on mobile device or on backend device.
Supervisor
i.e. medical personell
Backend Data Mining
optimising backend classification
e.g. by genetic algorithms

Figure 5.2: Visual concept of the MECoS system

5.2.2 Build a moving ECoS - MECoS

The requirement of a high-quality and very energy efficient and autonomous Mobile Care System requires the application of a very dynamic algorithm that is capable of adapting to every individual patient very "closely". A direct relationship between overall efficiency of the system (classification quality, energy consumption) and health state of the monitored patient can be drawn: as long as the health state of the patient is in a normal state, a very energy efficient and economic mode of operation can be selected (no communication with back-end systems, low sensor sampling rate, etc.), but as soon as the health state leaves normal boundaries (the definition of the term "normal" in this case is very specific to each individual patient), the efforts on the mobile device increase and economy is reduced. As soon as potential cases of emergency are identified, the analysis and classification of the sensor data cannot be performed by the mobile device itself - finally in these cases a wireless data communication line has to be opened and sensor data have to be transferred to a much more powerful back-end system for further processing and better analysis.

We can conclude that the most economic and efficient mode of operation of the mobile system is indirectly achieved by optimal classification of the physiological constitution of the individual patient, i.e. the core requirement of the mobile system is to permanently adapt itself to the individual patient as well as possible during operation by utilization of online machine learning concepts.

As explained above, an important result of literature research - in my opinion - is the fact that there is a need for a distributed online machine learning system in the mobile care context. As, in some cases, control has to be delegated to a more powerful (and therefore also smarter) back-end system, a way of distributed learning has to be possible. Somehow the system has to be able to transfer the acquired and/or the current knowledge from one node to another and vice versa.

Assumption: The resulting system has to be capable of extracting the current “mind” of the learning algorithm on one node, move it over the nework and import or inject it onto another node somewhere else. This ability of distributed learning on a shared single knowledge base will deliver best classification results at any point in time and provide high efficiency and autonomy for the resulting system.

During literature research a very promising concept called ECoS (Evolving Connectionist Systems paradigm, [95, 110, 101]) was found (see 4.3.10). It describes the characteristics of a very dynamic online learning system with attributes and characteristics that seem to perfectly match the requirements of this research. ECoS itself is not an existing system or a concrete implementation, ECoS is a concept. From my point of view the ECoS paradigm seems to be an ideal starting point for addressing the aims of this research.

ECoS itself is briefly presented below. My idea of how ECoS can be used as the foundation and the advancement of ECoS to a moving, online machine learning algorithm called MECoS - being perfectly applicable in the heterogeneous distributed system in the mobile care context - is shown in the following sections.

5.2.3 ECoS - the Evolving Connectionist Systems Paradigm

The underlying source of the above ideas seems to be represented by the theoretical "ECoS paradigm" concept, as presented by Prof. Nikola Kasabov[95, 110, 101]. Researched by Kasabov and others, several investigations and experiments are available in literature. They provide a profound basic knowledge of the characteristics, strengths and weaknesses of systems obeying the ECoS paradigm ([70, 102, 109, 108, 111, 103, 110, 95, 96, 71, 216, 214, 215, 213, 101, 104, 98, 97, 106, 105, 99, 100, 107]). Out of all the investigated and evaluated methods and technologies, in my opinion, ECoS seems to be the most promising path to follow as it offers room for possible answers to all the open questions.

Kasabov starts developing his ECoS principle by having a look at well-known problems of current systems and derives seven major requirements for modern, open connectionist based systems:

- learn quickly from a large amount of data (fast, single-pass training)
- adapt itself in real time in an on-line mode
- open structure where new features can be introduced at later stages
- accommodate everything that is, has been and will be in an incremental way
- be able to learn and improve through interaction with other COS (connectionist systems)
- adequately represent time and space in different scales

- be able to analyse itself (self-analysis and rule-extraction)

ECoS systems have to obey these rules and the previously shown requirements of Mobile Care Systems seem to fit these perfectly.

The ECoS (evolving connectionist systems) principle ([95, 110, 101]) describes the environment and requirements of online learning, connectionist based systems ("connectionist" means: network based). Apparently the ECoS paradigm (and its implementations) provides a technique for tackling the research challenges described: a computationally efficient, online-learning algorithm that can be based on an Artificial Neural Network system.

A valid description of the fundamentals of ECoS and the representative EFuNN (Evolving Fuzzy Neural Network) implementation can be found in [96]. It points out that a significant advantage of ECoS implementations like EFuNN is the incremental, local learning algorithm and also mentions that one of the problems is finding optimal values for the evolving parameters (sensitivity, error threshold, learning rate). EFuNN investigated in more detail can be found in [108]. This paper provides a good theoretical background for implementing ECoS/EFuNN yourself.

In [109] the authors compare two architectures of knowledge-based neural networks: FuNN and EFuNN. It points out that EFuNNs are much faster than traditional FuNNs (Fuzzy Neural Networks - a hybrid system that combines Fuzzy-Logic with ANNs) and therefore applicable to online learning.

The ECoS paradigm may seem to be rather outdated meanwhile, but as it is basically a theoretical concept ([95]) and not a concrete implementation it can always be applied to (or its applicability tested against) modern technologies. There is still ongoing research ([189, 98, 106, 107, 105, 19, 99, 100, 97]) and more and more applications of the ECoS paradigm to current AI approaches are becoming available. ECoS is a relevant paradigm for modern AI based systems, for instance eGNN - Evolving Granular Neural Networks from Fuzzy Data Streams presented by Leite, Costa and Gomide in[128] and its application to spiking neural networks called deSNN - dynamic evolving spiking neural network as presented by Kasabov et al in [106] and related publications ([98, 107, 99, 100, 97]) or the recent advancements of the original Evolving Neuro-Fuzzy Systems (EFuNN, [109, 108]) as presented by Kasabov in [105] show its current relevance.

5.3 From ECoS to MECoS - the Moving Evolving Connectionist System

5.3.1 Introduction

Based on the results so far, a concept for a system that addresses the research aims becomes visible:

- the aim is a highly efficient and autonomous mode of operation of a Mobile Care System
- simplicity of the mobile care device increases efficiency and autonomy

- a powerful back-end system shall be able to compensate for the shortcomings of the simple mobile device if necessary
- a distributed system concept which optimally balances this cooperation between mobile and back-end has to be found
- to perfect this balance, the system has to adapt itself to the individual patient during operation - online learning abilities are required
- if the mobile device makes a wrong decision (false positive) and the back-end system creates a different result, the knowledge base of all distributed nodes has to be updated ("learn") to avoid future occurrences of this mistake –> somehow the knowledge has to be shared, learning has to be performed in a distributed manner.

The MECoS Concept and ECoS The "enabler" for the presented MECoS approach is the ECoS paradigm of Prof. Nikola Kasabov ([95, 110, 101]): ECoS are able to represent time and space in a compressed way. This is done within their locally optimized clusters and hence the "mind" of an ECoS can easily be extracted and injected somewhere else. I simply call this technique "state serialisation": the knowledge or the mind is transferred into a more powerful "body" to cope with the problem. The clusters in EFuNNs [109, 108, 105] for example represent all input data that has ever been presented to the algorithm presented in the form of ECM (evolving clustering method) and what has been learned in the form of neurons and weights. This very compressed information really has the characteristic of a mind, it contains "memories" (all input data presented) and "behaviour" (represented by the decisions taken due to neurons and weights). This mind has to be serialized, transferred and respawned on another machine with more power and even more available input data. It is even possible to utilise additional input data sources on the back-end node, as ECoS based systems support changing the structure of the network structure during operation. This is shown later: within the performed experiments to test this approach, an assumed calculated statistical value was successfully integrated into the system as extra input data only available to the classifying algorithm at the back-end node.

In order to build a working prototype of the presented MECoS approach and to be able to test it, the following actions have been performed:

- **Explain a detailed concept of the MECoS System Idea**
  Based on the results of investigation of Mobile Care System issues and literature research, the idea of the MECoS System has been created. This idea was presented in the form of a written concept which describes the unique characteristics of this kind of system (as described in this thesis in section 5.2).

- **Provide a proof-of-concept implementation of MECoS System**

  - ✧ **Build a working implementation of an ECoS system**
    The basis for a MECoS System is ECoS. As no implementation of an ECoS based system is freely available for further usage (I have contacted Prof. Kasabov directly during my research and asked), an initial ECoS implementation of my own was

developed (called MECM - mobile evolving connectionist method). As a starting point the freely available sourcecode of a multilayer perceptron ([159, 183, 67, 161, 34, 41]) was used and was adapted to become a working ECoS algorithm implementation - advised by Prof. Kasabov using the descriptions and information provided in [110] (see section 6.3.1).

✧ **Implement a way for distributed sharing of the full knowledge base of the algorithm**

The algorithm was designed in a way that at every point in time the acquired knowledge can be extracted, transferred online and imported or "injected" somewhere else - e.g. from the mobile to the the back-end system and vice versa. I called this process simply "state-serialisation", the structure of the evolving network can be written to a file, transferred over the network and the network rebuilt somewhere else. First tests serialised the network structure in a human-readable text format but to optimise and reduce the amount of data that has to be transferred, the state-serialisation process was changed to produce a much more compact binary format subsequently (see section 6.3.2).

✧ **Allow a dynamic number of input parameters on different nodes**

The MECoS algorithm was designed in such way that the back-end system can utilize and integrate additionally available data sources as "dynamic additional input params". This should allow more exact classification by e.g. integrating results of data mining processes of large databases that are simply not available on the mobile devices. All this has to run on the identical knowledge base as the mobile classifier does (–> the back-end system can take learning steps which will also enhance the knowledge of the mobile classifier even if it is not able to learn this by itself).

- **Develop a Data-Generator**
  As it is very difficult and only possible with very high effort to get realistic/real physiological test data (because of ethical issues), a data generator was developed, that is capable of generating several datasets consisting of simulated physiological measures like pulse, heart rate, blood oxygen level, etc. The data series consist of normal variations and in some cases, a case of emergency has to be generated into the time series of test data (see section 6.2.3).

- **Define Performance Numbers that allow benchmarking**
  To allow benchmarking of optimisations of the algorithm prototype and also of two different approaches, common performance numbers that directly or indirectly relate to the resource- and energy consumption of the mobile device were defined and have been measured during simulation runs (see chapter 7).

- **Build a second approach for comparison**
  Furthermore, a standard client-server approach for comparison and testing the MECoS approach was built. This approach also has a cooperating mobile classifier and a back-end classifier, but with their independent knowledge bases on each device as state of the art

systems would look like. They do not have a shared knowledge base but the back-end system acts as a reinforcement learning supervisor for the mobile system (similar to a conventional distributed classification system).

- **Perform simulation runs and collect statistical data**
  Statistical data of the defined performance numbers and measures were finally collected in representative experiments using the generated test data. This allowed comparison of the performance of the two different approaches and therefore tested the hypothesis of this research project.

Chapter 6

Experiment Design

6.1 Introduction

THE presented MECoS approach perfectly fits into the problem domain of Mobile Care Systems. It is highly applicable and, compared to other systems, achieves better results by providing longer autonomous runtime of the system without additionally reducing the patient's mobility.

The idea of the MECoS approach is based on the results of literature research and the paradigms of ECoS. To be able to find out whether a working implementation can be achieved at all and to prove that the presented advantages really become apparent in practice, experiments in the form of simulation runs have to be conducted.

Because of ethical restrictions in the medical context it is very hard to perform tests with real patients or real physiological data of patients. To prove the advantages of the MECoS approach, the original source of the input data is not relevant - in my opinion it just has to be very similar to the physiological data of real patients, but the classifying algorithm basically works independently of the presented data semantics. Because of this reason a data generator was developed for the simulation environment that represents a classical mobile care situation with key functions.

Within this simulation environment comparative runs between a traditional client-server based Mobile Care approach and the MECoS prototype will be performed and the results will be compared to each other. This allows a realistic assessment if the MECoS approach and the underlying hypothesis can be tested as positive or not.

6.2 Experimental Setup

The aim of the experiments is to build and investigate a working prototype of the MECoS System and to test its superiority over traditional systems. The applicability of the classification results of the system and whether the presented deliverable objectives can be achieved with lower resource consumption needs to be investigated. Furthermore, the system's behaviour should also be analysed - compared to established approaches.

It is required to define a set of key numbers which are used to measure and compare the results of the experiments. Expected outcome of the simulation runs is a database containing these defined and measured numbers for every simulation run performed (see appendix A). The number of performed simulation runs should be relatively high, providing as many different scenarios (a scenario is represented within the generated physiological datasets) as possible to depict the behaviour of the MECoS approach as precisely as possible.

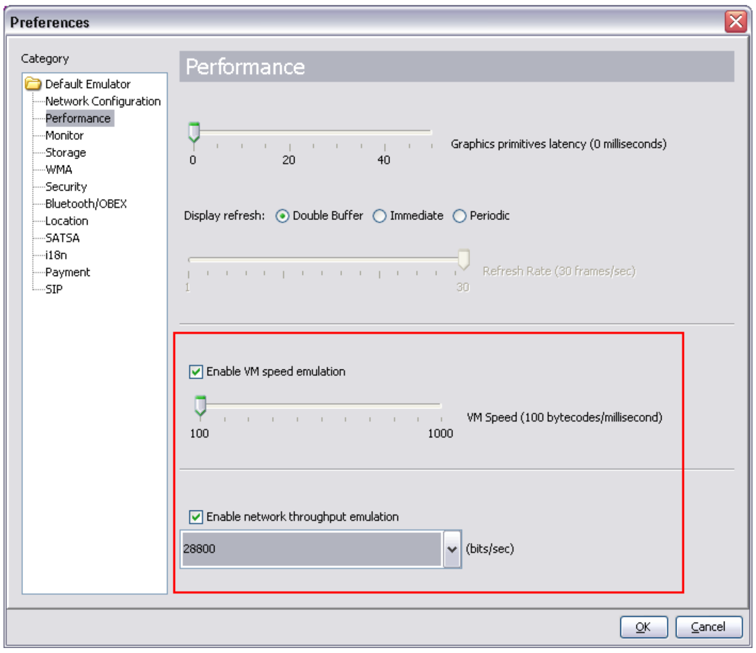
Finally, the presented hypothesis should be tested positively (or even negatively) by analysis and evaluation of the experiment results (results are presented in chapter 7).

Because of the prior knowledge of the author, the Java Development Kit (JDK) [6] was chosen as the development platform for the experiments and the building of the simulation environment. The JDK provides numerous freely available tools for the intended use within this research, a key tool for instance is the Java Wireless Toolkit (WTK)[7] which provides a comprehensive development tool for simulating a restricted mobile device.

6.2.1 Mobile Device

For the simulation of a restricted mobile device the Java Wireless Toolkit (WTK)[7] is used. The provided mobile device emulator of the WTK is able to simulate reduced processor speed, slower network throughput and memory access of a mobile device and also provides a means of measuring performance and resource consumption which is ideally applicable to the experiments.

The following picture shows configuration options of the WTK, highlights the speed and network throughput emulation within the red box:



Graphics primitives latency (0 milliseconds)

Display refresh: [✓ Double Buffer] Immediate Periodic

Refresh Rate (30 frames/sec)

[✓] Enable VM speed emulation

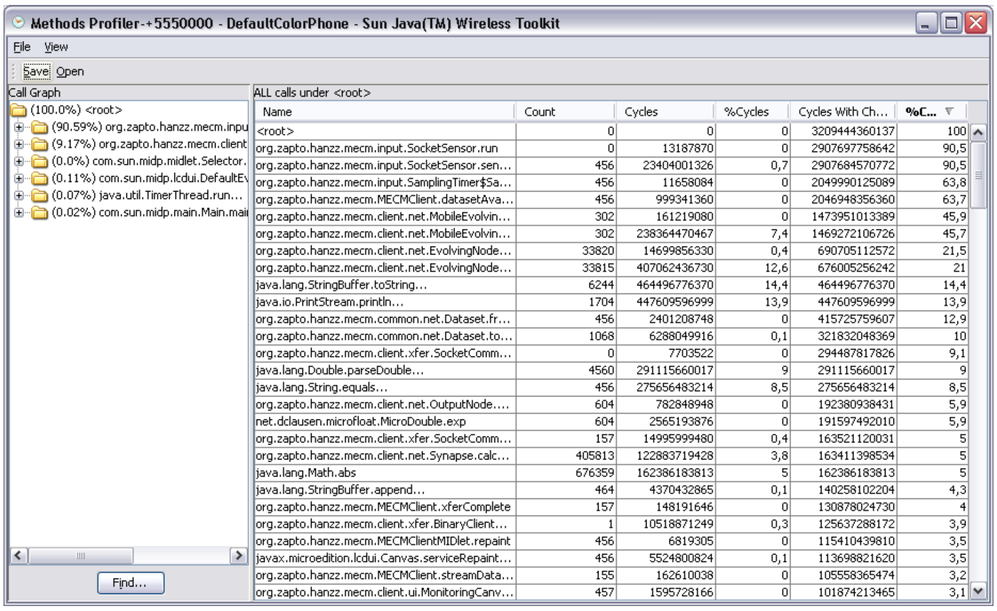
VM Speed (100 bytecodes/millisecond)

[✓] Enable network throughput emulation

28800 (bits/sec)

Figure 6.1: Simulation of reduced mobile resources within the Wireless Toolkit

The following graphic shows the so-called “method profiler” of the WTK, which allows the measurement of the number of method invocations and especially the number of used processor cycles of a specific method during a complete experiement run. This tool is very useful, because resource consumption of specific methods can be measured without distortion.


Methods Profiler-+5550000 - DefaultColorPhone - Sun Java(TM) Wireless Toolkit
File
View
Save
Open
Find...

Call Graph

- (100.0%) <root>
  - (90.59%) org.zapto.hanzz.mecm.inpu
  - (9.17%) org.zapto.hanzz.mecm.client
  - (0.0%) com.sun.midp.midlet.Selector.
  - (0.11%) com.sun.midp.lcdui.DefaultEv
  - (0.07%) java.util.TimerThread.run...
  - (0.02%) com.sun.midp.main.Main.mai

ALL calls under <root>

Name	Count	Cycles	%Cycles	Cycles With Ch...	%C...
<root>	0	0	0	3209444360137	100
org.zapto.hanzz.mecm.input.SocketSensor.run	0	13187870	0	2907697758642	90,5
org.zapto.hanzz.mecm.input.SocketSensor.sen...	456	23404001326	0,7	2907684570772	90,5
org.zapto.hanzz.mecm.input.SamplingTimer$Sa...	456	11658084	0	2049990125089	63,8
org.zapto.hanzz.mecm.MECMClient.datasetAva...	456	999341360	0	2046948356360	63,7
org.zapto.hanzz.mecm.client.net.MobileEvolvin...	302	161219080	0	1473951013389	45,9
org.zapto.hanzz.mecm.client.net.MobileEvolvin...	302	238364470467	7,4	1469272106726	45,7
org.zapto.hanzz.mecm.client.net.EvolvingNode...	33820	14699856330	0,4	690705112572	21,5
org.zapto.hanzz.mecm.client.net.EvolvingNode...	33815	407062436730	12,6	676005256242	21
java.lang.StringBuffer.toString...	6244	464496776370	14,4	464496776370	14,4
java.io.PrintStream.println...	1704	447609596999	13,9	447609596999	13,9
org.zapto.hanzz.mecm.common.net.Dataset.fr...	456	2401208748	0	415725759607	12,9
org.zapto.hanzz.mecm.common.net.Dataset.to...	1068	6288049916	0,1	321832048369	10
org.zapto.hanzz.mecm.client.xfer.SocketComm...	0	7703522	0	294487817826	9,1
java.lang.Double.parseDouble...	4560	291115660017	9	291115660017	9
java.lang.String.equals...	456	275656483214	8,5	275656483214	8,5
org.zapto.hanzz.mecm.client.net.OutputNode....	604	782848948	0	192380938431	5,9
net.dclausen.microfloat.MicroDouble.exp	604	2565193876	0	191597492010	5,9
org.zapto.hanzz.mecm.client.xfer.SocketComm...	157	14995999480	0,4	163521120031	5
org.zapto.hanzz.mecm.client.net.Synapse.calc...	405813	122883719428	3,8	163411398534	5
java.lang.Math.abs	676359	162386183813	5	162386183813	5
java.lang.StringBuffer.append...	464	4370432865	0,1	140258102204	4,3
org.zapto.hanzz.mecm.MECMClient.xferComplete	157	148191646	0	130878024730	4
org.zapto.hanzz.mecm.client.xfer.BinaryClient...	1	10518871249	0,3	125637288172	3,9
org.zapto.hanzz.mecm.MECMClientMIDlet.repaint	456	6819305	0	115410439810	3,5
javax.microedition.lcdui.Canvas.serviceRepaint...	456	5524800824	0,1	113698821620	3,5
org.zapto.hanzz.mecm.MECMClient.streamData...	155	162610038	0	105558365474	3,2
org.zapto.hanzz.mecm.client.ui.MonitoringCanv...	457	1595728166	0	101874213465	3,1

Figure 6.2: Measuring used processor cycles within the Wireless Toolkit

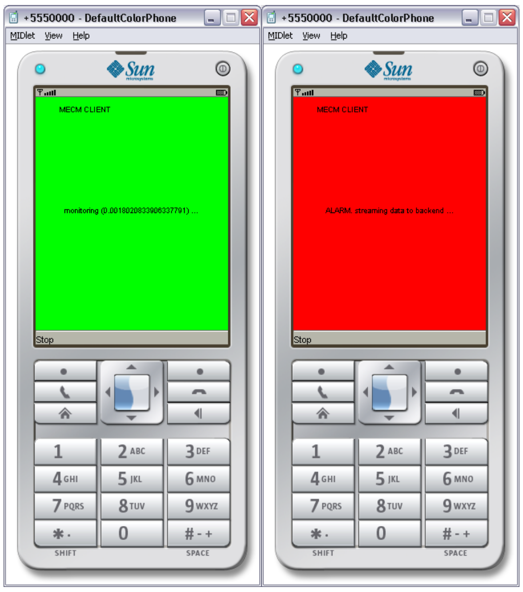

+5550000 - DefaultColorPhone
MIDlet
View
Help
MECM CLIENT
monitoring (0.0018020833906337791) ...
Stop
+5550000 - DefaultColorPhone
MIDlet
View
Help
MECM CLIENT
ALARM. streaming data to backend ...
Stop

Figure 6.3: Snapshot of the mobile MECM classifier during an experiment running in the Wireless Toolkit device emulator

Figure 6.3 shows screenshots of the developed MECM (Mobile Evolving Connectionist Method Classifier: the mobile implementation of the MECoS

Prototype) during an experiment run. The first screenshot with the green screen shows the classifier in "idle" monitoring state, i.e. the presented physiological input datasets are classified as normal (no-alarm), the system works as 'energy-savingly' as possible. The second screenshot with the red screen shows the classifier in "alarm" state, i.e. the input data has been classified as a possible alarm, delegation of control to the backend system has been performed: wireless data connection to the back-end system was opened up, the classifier was serialised and transferred over the dataline to the powerful back-end system and the mobile device now transfers all input data to the now leading back-end system.

6.2.2 Back-End System

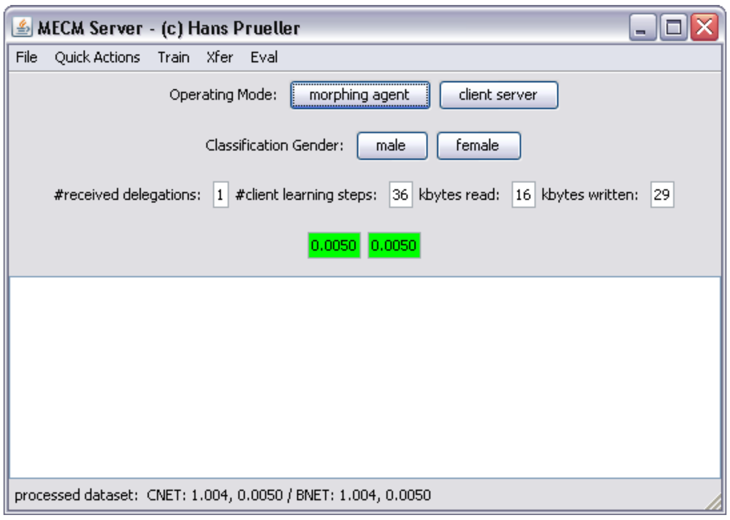
A major advantage of the back-end system compared to the mobile device is the availability of additional resources. These additional resources have been integrated into the simulation environment as well: On the back-end side, additional input data to the classifier are available, e.g. anonymised historical medial databases from other patients, statistical results from long-running data-mining tasks, and so on.

I expect that it can be taken for granted that the back-end system provides additional data sources to the classifier, so additional input nodes are applied to the backend-side MECoS structure. In the prototypical implementation for the experiments, this additional input is represented as a single given value that is set and changed manually within the test runs: the additional input data represents an assumed calculated probability of the current situation being an emergency, based on the gender of the patient and the outcome of an assumed data-mining analysis of additionally available back-end databases. Even if this is not very realistic, it represents an important proof-of-concept of integrating additional information as input data to the classifier on the back-end, to achieve higher classification quality than it would be possible on the mobile monitor itself (this is where the back-end system can identify a false positive and overrule the decision of the mobile system). This proves that the MECoS approach can deliver results of higher quality than a "client-only" approach.

The screen in figure 6.4 shows the control and status panel of the back-end system monitor implementation, called in short: "MECM Server" (Mobile Evolving Connectionist Method Server, which has been the basis for the MECoS System prototype). It shows that the control panel provides buttons for two different operating modes: "morphing agent" and "client server":

- morphing agent: in earlier phases of this research, MECoS approach was named "morphing agent" - operating mode "morphing agent" means MECoS features are activated
- client server: turns off MECoS features and the mobile and backend nodes run with their own classifier instances without the MECoS feature of a moving classifier with a single, shared knowledge base

This possibility to set different operating modes was used to perform comparable experiments with either the MECoS features activated or with a traditional client-server based logic. Switching the system from MECoS to client server mode, results in two independent online learning instances on the mobile and on the back-end side, with their own knowledge bases delegating control to each other instead of a single "mind" that moves from node to node as is the case in the MECoS system approach.



Operating Mode: morphing agent

Classification Gender:

#received delegations: 1 #client learning steps: 36 kbytes read: 16 kbytes written: 29

0.0050 0.0050

processed dataset: CNET: 1.004, 0.0050 / BNET: 1.004, 0.0050

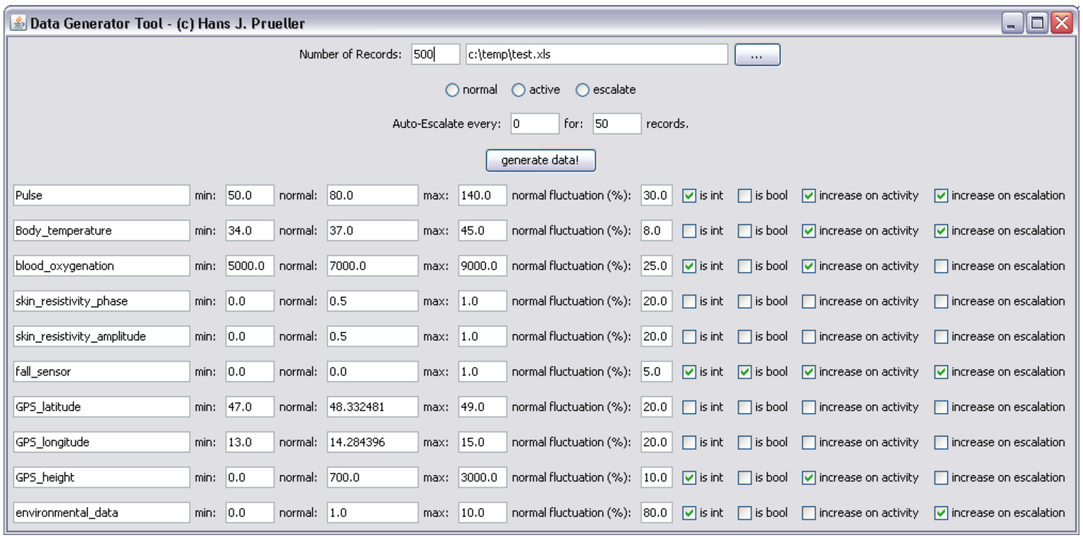
Figure 6.4: Back-end node MECM system status panel

6.2.3 Input Data

Medical data is underlying ethical restrictions and hardly available for research purposes (or only under very high restrictions). To investigate the proposed MECoS system and test the stated hypothesis, it is not relevant to perform the experiments with physiological data of real human beings. At this stage of research, it has to be tested whether the presented MECoS approach performs better than traditional mobile monitoring algorithms. The better performance is measured in less computing and resource efforts of the classification algorithm. The important thing to be able to compare

MECoS' performance to other approaches is that they are fed with identical input data to get comparable results. Of course, at later stages of research experiments with real medical data should be performed.

As an approximation of physiological input data will do, a data generator for exactly this task has been developed for the experiments. The generator is able to produce random datasets of different width and in different "dynamic characteristics", based on specified generation parameters as visible in figure 6.5. The generated data is designed to mimic real-world situations, e.g. they represent plausible values and curves of blood-oxygen level, heart rate, pulse, body temperature, skin resistivity and some more. The dynamic characteristics of the generated data can also be configured to simulate various scenarios for the experiments: a patient in normal state, an active patient e.g. simulating walking up a hill and ones in case of emergency where the physiological parameters escalate. The parameters dialog to configure the generator is shown in figure 6.5. The (important parts of)source code of the generating algorithm can be found in the appendix (see appendix B).



Data Generator Tool - (c) Hans J. Prueller

Number of Records: 500 c:\temp\test.xls

normal active escalate

Auto-Escalate every: 0 for: 50 records.

generate data!

	min:	normal:	max:	normal fluctuation (%):	is int	is bool	increase on activity	increase on escalation
Pulse	50.0	80.0	140.0	30.0	✓		✓	✓
Body_temperature	34.0	37.0	45.0	8.0			✓	✓
blood_oxygenation	5000.0	7000.0	9000.0	25.0	✓		✓	
skin_resistivity_phase	0.0	0.5	1.0	20.0				
skin_resistivity_amplitude	0.0	0.5	1.0	20.0				
fall_sensor	0.0	0.0	1.0	5.0	✓	✓	✓	✓
GPS_latitude	47.0	48.332481	49.0	20.0				
GPS_longitude	13.0	14.284396	15.0	20.0				
GPS_height	0.0	700.0	3000.0	10.0	✓		✓	
environmental_data	0.0	1.0	10.0	80.0	✓			✓

Figure 6.5: Data Generator Tool for the experiments

Every experiment was performed with a dataset width of 10 (i.e. 10 simulated physiological or environmental input values) and in most cases over 1000 randomly generated sample datasets were presented within each test run. To allow a comparison of the results, every sequence of sample data sets was saved into a file and absolutely identical data sets in identical sequences were presented once to the MECoS and once to CS (client server) approach.

A representative view on the input datasets as generated by the data generator tool and used within the simulation runs can be found in the annex of this thesis (see appendix C).

Input Dataset

In all experiments that were performed, the generated dataset consisted of the following values. The aim was to build an applicable set of real-world like input measures used by the classifier to determine the patient's health state:

- Pulse (value range between 50 and 140)
- Body Temperature (decimal value range between 34.0 and 45.0)
- Blood Oxygen Level (value range between 5000 and 9000)
- Skin Resistivity Phase (decimal value range between 0.0 and 1.0)
- Skin Resistivity Amplitude (decimal value range between 0.0 and 1.0)
- Fall Sensor (boolean value true or false)
- GPS latitude (decimal value range between 47.0 and 49.0, i.e. movement of the patient to different places)
- GPS longitude (decimal value range between 13.0 and 15.0, i.e. movement of the patient to different places)
- GPS height (value range between 0 and 3000, i.e. movement of the patient to different places)
- environmental data (assumed value representing environmental circumstances, range between 0 and 10 e.g. encoded value representing temperature, air-pressure, humidity, etc.)

The physiological input values like pulse, body temperature, blood oxygen level etc. are combined with environmental input values. Although environmental conditions are not directly related to the physiological constitution of the patient, they have some influence because the physiological condition also depends on the current context of the patient: where is he or she located and how are the conditions there? Hence, the current position as GPS coordinates, and for the experiments an abstract “environmental data” value for meteosensitive patients has been integrated (e.g. like a barometer value or something similar that can be used as an input value for meteosensitive patients). As mentioned above, a representative view of the input datasets is located in the annex of this thesis (see appendix C).

Input Databases and Reducing Error

As mentioned in section 6.2.3, the generated input data was saved into several re-usable database files to provide identical dataset sequences presented to the two algorithms compared in different experiments. To reduce statistical error, 24 different databases - each containing 1000 datasets with different characteristics - were generated. This allowed the performance of a relatively high number of experiments to test the MECoS approach and to simulate various situations, e.g. an idle patient, a very active patient, borderline cases and escalation cases, etc.

6.2.4 Comparison to a classical approach

To measure, quantify and compare the outcome of the experiments, a second system based on a traditional client-server approach was implemented too: an online learning algorithm on the client side and an online learning algorithm with additional input on the back-end side (to provide circumstances similar to the MECoS prototype), but with one major difference: the client-server approach works without the "moving" part: the knowledge is not serialised and moved between mobile and back-end node in case of an identified alarm, each node learns on its own knowledge base as is common in current multiagent systems: The back-end only acts as a supervisor to the mobile node and communicates whether the triggered alarm was a true or a false positive one. All learning of the mobile nodes classifier also takes place on the mobile node itself. For the experiments, this was achieved by simply adding an "operating mode" parameter to the existing MECoS software which then worked without activating the MECoS key features. See also section 6.2.2.

6.2.5 Performance Numbers

As the MECoS and the traditional Client Server (CS) approach simulation runs results have to be compared, the following key numbers based on the measured values have been defined and used to evaluate the results:

- ProcCycles: Number of Processor Cycles used (overall efforts for processing the full range of datasets presented during an experiment)
- ClassTime: Average Classification Time per Dataset on client
- CommOpened: Number of Connections to the back-end system that were opened during a simulation run (very costly, [208, **?**])

- CommActive: Connection active (active data transfer) time
- Data Xfer: bytes sent/received; overall number of bytes transferred
- Net Size: number of nodes in classifier network structure

These numbers were chosen because they all indicate complexity and resource consumption of the algorithm on the mobile device, being directly related to energy consumption and autonomous runtime of the system. This allows direct implication regarding meeting the aims of the MECoS approach: to achieve minimal resource consumption and maximum autonomous system runtime compared to other approaches.

6.3 The MECoS Prototype

The ECoS concept is presented as a theoretical construct in literature. Although several concrete implementations of online learning algorithms obeying the ECoS paradigm are being presented, e.g. EFuNN ([109, 101, 111, 212]), the source code is not publicly available for any of these implementations. Hence, a new and independent ECoS implementation based on the ECoS paradigm (as presented in [110]) and on a multilayer perceptron [67, 159, 183, 161, 34, 41] for the application within the mobile care context has been developed within this research project. The implementation of the MECoS prototype has been divided into several sub-tasks and milestones. Every time a milestone could be achieved successfully, the next higher level was targeted. This allowed an iterative step-by-step approach to building a working prototype of the MECoS concept.

6.3.1 Building an Evolving Network

The initial project was named “MECM”: mobile evolving clustering method, an incremental locally optimising learning algorithm according to the ECoS paradigm. Because of the author’s previous knowledge, the implementation used the Java platform ([6]), based on several publicly available Java-implementations of a multilayer perceptron [8, 9]. The learning ability of the algorithm could be proved with simple examples, for instance the base algorithm was able to learn the logical operators AND, OR and X-OR (3 input values: valueA, valueB, opCode; 1 output value).

As described above, the underlying neural network of the MECoS prototype was based on a Multilayer Perceptron and the description of the Evolving Connectionist Systems paradigm by Prof. Kasabov in [110]. The main difference of evolving connectionist systems related to traditional Artificial Neural Networks is, that they do not have a static structure (after training). ECoS based systems adapt themselves in real time and can also change their structure by building new nodes (or also called clusters) during runtime. This results in a very dynamic system which perfectly fits into the requirements within the Mobile Care context. The nodes of the connectionist structure, the clusters, are nothing but a point in a coordinate system with a radius (within a network with 2 dimensions). The classifier tries to classify presented datasets into existing clusters which can be adjusted, or new clusters can be created. The search for the best fitting cluster is performed by calculation of the euclidean distance of the arrived dataset to each existing cluster as shown in listing 6.1.

Algorithm 6.1 Calculation of euclidean distance of a dataset to an evolving cluster

```
private double calculateNormalisedEuclideanDistance(Dataset ds,

{
                /* |a| = (a1^2  + a2^2) ^ 1/2 */
                double x1 = ds.getX();
                double y1 = ds.getY();
                double x2 = c.getX();
                double y2 = c.getY();
                /* a2+b2=c2 ... a1^2 + a2^2 = a^2...
        * a = sqrt ( a1^2 + a2^2 )
        * a1 = x2 - x1
        * a2 = y2 - y1  */
                double a1 = Math.abs(x2-x1);
                double a2 = Math.abs(y2-y1);
                double a1q = Math.pow(a1,2);
                double a2q = Math.pow(a2,2);
                double a = Math.sqrt(a1q+a2q);
        return a;
}
```

Listing 6.2 shows a further important part of the algorithm which brings the dynamic characteristic into the neural network based structure, i.e. where it performs the clustering for a newly arrived dataset: if a fitting existing cluster for the currently arrived dataset is found, it is used (and probably adjusted). If not, a new cluster is created.

Algorithm 6.2 Net adjusting existing or creating a new cluster

```
if(nearestCluster.getRad() > nearestDistance)
{
        //the dataset belongs to nearest cluster as it is
        // within its "circle"
        return;
}
else
{
        //we have to find a possibly fitting cluster
        final Cluster candidate = findPossiblyFittingCluster(ds);
        if(candidate==null) {
                //the dataset does not belong to an existing cluster
                //and also no possibly fitting cluster by adjusting
                //its center and radius exists-create a new cluster
                createNewCluster(ds);
        }
        else {
                adjustCluster(candidate,ds);
        }
}
```

Further, a tool to visualise the clusters created by the learning algorithm was developed. A problem with this first implementation was the correct interpretation of the linguistic description of the algorithm in [110] and correct translation into sourcecode of a programming language. The visualisation tool helped to understand what evolving clustering does (this is what local learners do, building locally optimised clusters - see also [144]). An example of a test run of the Mobile Evolving Connectionist Method (MECM) and the visualisation of the created clustering (pink circles) of the presented input data sets (blue points) is shown in figure 6.6.

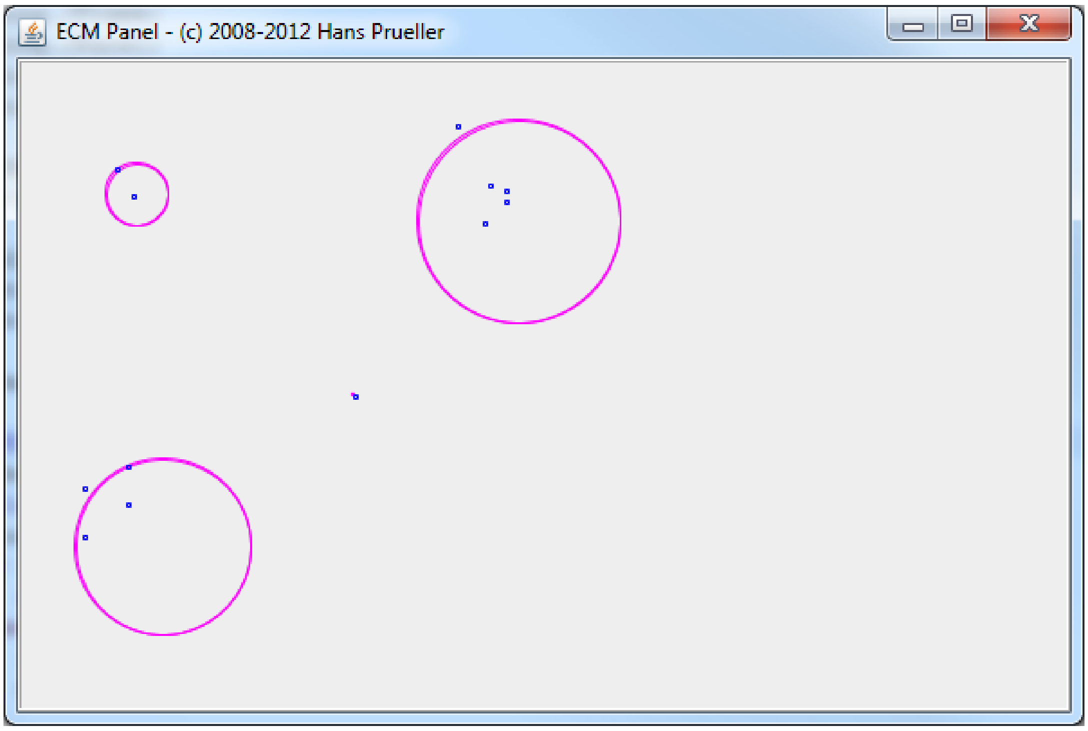


Figure 6.6: Visualisation of Evolving Clustering Method

After this very simplified proof-of-concept implementation of an ECoS-based online machine learning algorithm "MECM", the development of a complete prototype for the MECoS Mobile Care System prototype was performed, but with a little change: a ANN based classifier is required to achieve best results in patient monitoring, MECM was not a real ANN but rather reinforcement-learning-based and just a proof-of-concept for the evolving clustering part. For the MECoS prototype, a full multilayer perceptron (which is an ANN implementation) was used as a base and extended and adapted successively.

6.3.2 Neural Network Serialisation

In the first prototypes, the serialisation of the Neural Network (i.e. the mind of the system) was performed in a very simple, text-based way. This facillitated basic proving that the knowledge of an ECoS-based system (i.e. the mobile node) can be extracted, transferred and injected from one device into another ECoS-based system instance (i.e. the back-end node) even if there is a different network structure (the back-end system has additional input parameters to achieve better qualification results).

After the initial milestone could be proved, the serialiser was converted from text-based mode of operation to producing a binary format [196], which is much more efficient. The efficiency of the serialiser algorithm could be improved but not as expected, the net size did not shrink significantly as a performed test run showed (table 6.1).

	Time Duration	Header Size	Net Overall Size
Text/ASCII	6750 msecs	86 bytes	8630 bytes
Binary	3250 msecs	36 bytes	8512 bytes

Table 6.1: Text-based vs. Binary Serialiser Efficiency

Further optimisations using net state compression were performed. Compression has to be a consideration of performance (CPU usage) and compression rate. For initial tests a very simple run-length-enconding [172, 18, 35, 117] based compressor was implemented, which simply compresses a sequence of bytes to reference bytes and a counter. (e.g. instead of transferring a byte value of 197 five times, the byte value 197 is transferred once and a counter of 5).

An implementation of an alternative compression algorithm has been found (range coding compression)[10], which has a better compression rate but also requires more computation and therefore has less performance. At least on the weak mobile device this could lead to system performance problems. For the initial experiements, the already implemented run-length-encoding based compression algorithm was used. To get a first impression of the overall performance of the MECoS prototype and to be able to compare it to the classical client-server based approach this was sufficient. For further research, the implementation of better or alternative compression algorithms like the found range coding compression algorithm should be performed.

6.3.3 Building the Simulation Environment

The above sections have presented the important key-tasks for building a working prototype of the MECoS concept and being able to perform experiments using this prototype. Of course many more activities had to be performed to build up a complete and fully working simulation environment in which the experiments could be conducted. The following paragraphs give a short overview of those activities (including the prototype and required additional tools for performing the simulation runs and collecting the results). In order to build the simulation environment for the MECoS approach, the following tasks were performed (in this sequence):

Implement an ANN-Test

- implement non-evolving, non incremental variant of ANN in Java as a base (from scratch, the MECM test was not re-used)
- get deeper understanding of how ANNs really work by building an implementation myself (practical experience is required to fully understand the topic)
- look for current java implementations of ANNs and investigate them
- a problem was the application of correct activation-functions and parameter values (momentum, learning rate, etc.) of the network to achieve usable learning results

Implement the EMLP (evolving multilayer perceptron) algorithm

- build evolving multilayer perceptron as a first minimal ECoS implementation of the previously built "static" ANN by enhancing/changing the algorithm to an evolving one as described in [110]
- like ecmtest translating the linguistic description of ECoS working scheme into a Java programme code was different
- finding good parameters to get a working emlp is quite difficult, too: even if the algorithm is basically working, parameter tuning is a time consuming task

Implement emlpME (emlp java microedition test)

- migrate result of emlptest to mobile java platform that can run on the Java WTK emulator which emulates slow speed and network connection of mobile/portable devices
- problem: reduced API on Java ME: no full floating point operations support, had to look for replacement for several Math functions that are required (Math.exp, Math.random, ...)
- after research on the internet: “microfloat package” [11] was found and integrated, provides a replacement for missing floating-point operations by long/double conversion

Prove serialisation and de-serialisation of the emlp (ME)

- the working emlp (and emlpME for the mobile device) network structures were serialised into a file and de-serialised again (based on algorithms as described in [196]), i.e. the transfer of the knowledge base over a dataline was tested (proof of concept) and could basically be proved as the serialised and de-serialised network produced exactly the same results when classifying a dataset sequence.

Consideration of further steps

- as the development of a simple working ECoS system from scratch succeeded (emlp and emlpME) and also the serialisation and de-serialisation could be proved as well, the former steps can be seen as “proof of concept milestone (1)", because it proves the development of an ecos that can be "moved" via state serialisation.

- after this milestone had been achieved, a short consideration of the required next steps was performed:
  - ✧ the next step is to build an unbalanced system simulation, with the classifier moving from one side to the other, which represents the second proof-of-concept milestone (2).

Completion of the simulation environment

- a complete simulation environment is required in which the above systems can be tested "fully integrated" and where the defined numbers (as listed in section 6.2.5) can be measured
- the components from milestone 1 were taken and integrated into a fully working simulation environment to start the experiments:
  - ✧ two nodes to be simulated are required: mobile and back-end/server were implemented
  - ✧ mobile was run as emulated mobile device within the java wireless toolkit (Java WTK) as the WTK is able to emulate the slow processor speed and network bandwidth of mobile devices
  - ✧ the server was run as standard java application on a PC
  - ✧ the data generator produced several different scenarios for the simulation runs, each scenario containing about 1000 records

saved into an XLS file to reproduce simulation runs with identical datasets (required to compare MECoS to traditional approaches)

6.4 Summary

Based on the author's previous knowledge, the Java-Platform [6] was chosen as the development environment for the experiments. The Java-Platform also delivers a practical tool, the Java Wireless Toolkit [7], which is used for mobile application development and is capable of simulating the limited resources of mobile devices. The back-end modules have been implemented as a standalone Java application, communication between mobile and back-end nodes is performed via standard TCP/IP sockets.

As there is no implementation of an ECoS-based system publicly available, an ECoS implementation based on the concepts as described in [110] had to be built (as suggested after Prof. Kasabov had been contacted directly). The self-implemented ECoS was built upon a freely available Java implementation of a standard multilayer perceptron. As initial tests of the self-implemented ECoS-paradigm-based online machine learning algorithm were promising, a complete simulation environment for the conduction of experiments was built upon this initial prototype: the MECoS system. During the development of the simulation environment it was considered right from the start, that the classifier had to run in different classification modes. The simulation environment has to be able to run different algorithm types for comparison: the presented MECoS approach and a traditional, static client-server based approach (CS). Every simulation run was performed under identical conditions in MECoS and also CS mode to compare the performance of those two approaches by the conducted measurements.

Conducting experiments with physiological data from real patients is affected by ethical restrictions and brings up a lot of bureaucracy. It is also very time-consuming and expensive and in some cases even impossible to do. As the specific use case of mobile care is not really relevant for the MECoS classifier, the experiments were conducted with artificially generated patient data but with the aim of being as realistic as possible. Although, of course, the generated data cannot be compared to real physiological input data measured by sensors attached to a human being, the MECoS approach learns the individual characteristics of this generated data - similar to the fact that every human patient is individually different and "delivers" individually different physiological measurements to the algorithm that has to learn it and adapt itself to it.

A data generator tool was developed within this research project to generate masses of physiological input data time series in different "scenarios". The datasets consist of realistic value ranges of e.g. blood sugar, body temperature, blood oxygen level etc. The scenarios can be chosen for each time series that is being generated and saved as a database. A scenario defines the characteristic of the curves of the input data measures, e.g. normal, active/sport, emergency.

Based on various generated databases, the experiments could be conducted and compared to identical input data sequences as often as required: compare different algorithm types results, refine the algorithms, etc. To be able to measure and compare the outcome of the experiments, some key numbers have been defined which represent the resource intensity or economic mode of operation of the algorithm.

The results are presented and discussed in the following chapter.

Chapter 7

Results

7.1 Introduction

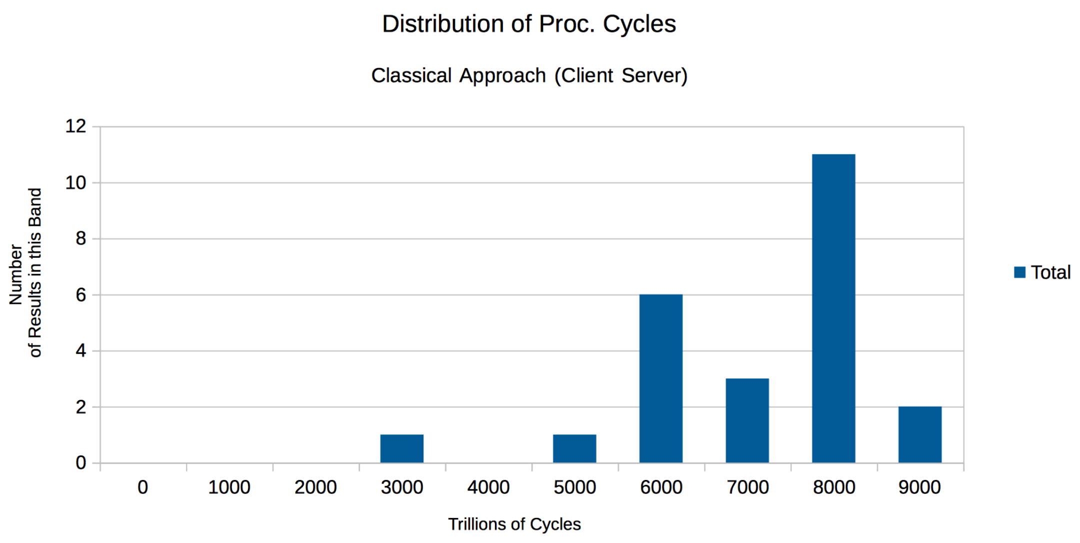
THE aim of this research is to provide an efficient distributed online machine learning algorithm that is applicable within heterogeneous distributed systems and performs better than traditional (client-server based) approaches in terms of resource and energy/battery consumption.

Experiments have been conducted running the prototype of the presented MECoS approach and for comparison with a classical client-server (CS) based approach. The experiments have been performed within absolutely identical environments (same hardware, same input data).

From the numerous measurements collected during the experiments, a set of important key numbers was selected and analysed to derive metrics and compare the two different algorithms.

The results show significant improvement with the MECoS approach, it results in an average reduction of Processor Cycles used by about 30%, and an 12% improvement in required communication efforts.

A visual presentation of outcome and evaluation is provided by the following histograms: The first two compare the distribution of the overall number of processor cycles used (see figure 7.1). The shift of the bars to reduced processor cycles usage of the MECoS approach compared to the CS approach can be observed clearly:



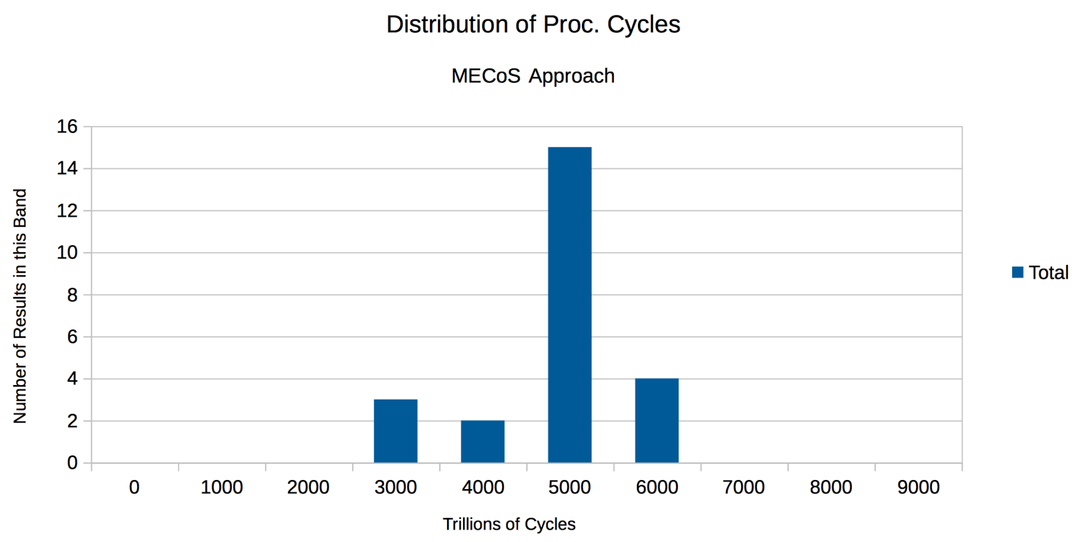
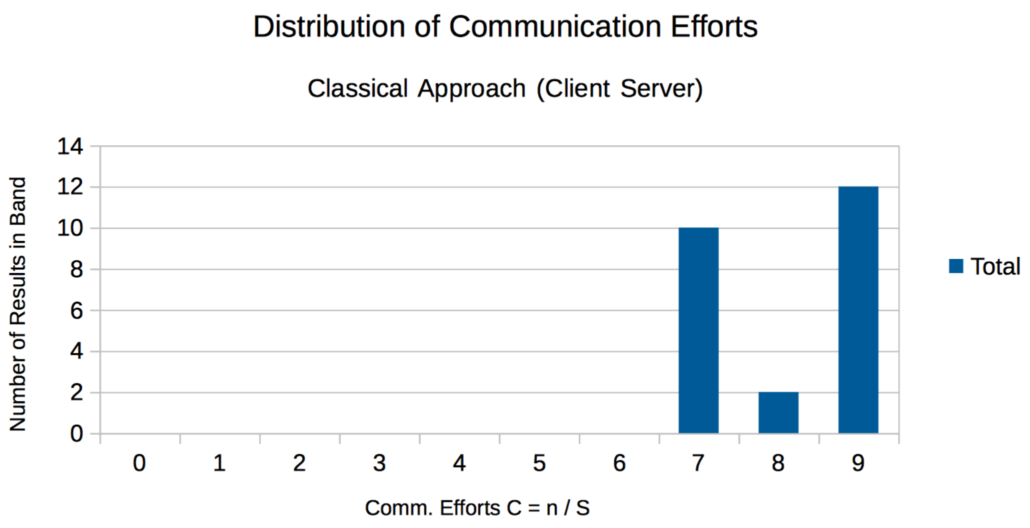


Figure 7.1: Histograms of Processor Cycles Used - CS compared to MECoS

The second major issue related to efficiency and resource consumption are the required communication efforts (sending and receiving data is very expensive for mobile devices). The following two histograms visualise the distribution of required communication efforts of the MECoS approach compared to the traditional Client Server approach (see figure 7.2). The distribution, which mainly depends on the characteristics of the generated input data, is not as smooth as with processor cycles. But the shift of the bars to significantly reduced overall communication efforts (C = n / S, where n is the number of connections that were opened and S is the amount of data that was transferred) of the MECoS approach compared to the Client Server (CS) approach can be observed, too:



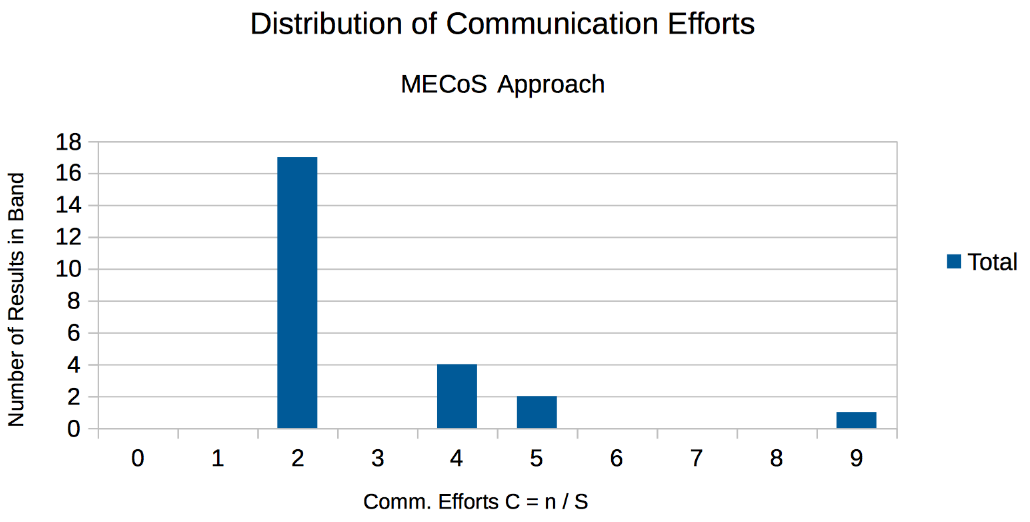


Figure 7.2: Histograms of Communication Efforts - CS compared to MECoS

The following sections present the results of the experiments more closely. The values investigated and the numbers used to compare the approaches are derived, the measures of the important numbers are explained and evaluated in more detail.

7.1.1 Values Investigated / Metrics

As the hypothesis has stated, the aim is to find a more efficient distributed online learning algorithm so the values that had to be measured within the experiments were mainly performance/efficiency and energy-consumption-related:

Measuring Energy Efficiency

Energy-related metrics have been derived from the basic equations of Martin et al [139] in chapter 3 as:

$Elocal = (PprocActive + PcommIdle) * Tsystem$

and

$Eremote = n * PactivateComm + (PprocIdle + PcommActive) * Ttrans + (PprocIdle + PcommIdle) * Twait$

whereas

- E$local$ ist energy consumption for local processing,
- E$remote$ is energy consumption for remote processing.

- n is the number of remote processing steps (i.e. how often a connection to the back-end system has to be opened)
- P$procActive$ is power consumption for active data processing
- P$procIdle$: power consumption of idle data processing
- P$commIdle$ is power consumption of idle communication
- P$commActive$ is power consumption of active data transfer
- P$activateComm$ is power consumption peak when communication is opened up
- T$system$ is the duration of local data processing
- T$trans$ is duration for data transfer tasks
- T$wait$ is duration for waiting for results of the remote processing

Constant factors specific to each hardware are the power-consumption numbers and processor capabilities, therefore the numbers of interest for measuring energy efficiency independent of the hardware specific factors are:

- T$system$
- T$trans$
- T$wait$
- n

but this is still only representative for a specific hardware infrastructure:

- $\mathrm{T}trans$ = S/r (size / ratio) - as data transfer rate (ratio) is strongly dependent on hardware, it can be treated as a constant value and disregarded, so the most representative variable of $\mathrm{T}trans$ can be derived as: S - amount of data that has to be transferred
- $\mathrm{T}system$: duration for active data processing - in analogy to the above paragraph, $\mathrm{T}system$ can also be refined and simply replaced by number of processor cycles W (as $\mathrm{T}system$ = number of processor cycles * time_duration_per_cycle and the time duration per cycle is a hardware specific constant value)
- $\mathrm{T}wait$: can be treated as "given" external constant value -> therefore it can be disregarded

The extraction of the relevant numbers from the above equations results in three values: number of processor cycles W, number of connections opened n and number of bytes that have to be transferred S. The number of processor cycles clearly relates to the processing efficiency, but to express the communication efficiency with a single number we need to relate S to n - based on the energy consumption characteristics of GSM/UMTS modems [38, 208, 64, 4]. As described earlier (chapter 3) it is better to transfer more data over fewer connections that have to be opened than to transfer less data over more connections, e.g.:

- sending/receiving 100 bytes over 1 connection that has to be opened up is better than:

- sending/receiving 40 bytes but over 2 required connections

To stay hardware independent within this research we directly relate S to n to express communication efforts. It is clear that this relation changes slightly because it depends on the energy-consumption characteristic of a specific hardware product, however, I think this does not change the outcome of the experiments comparing MECoS approach to other approaches in a major way and can be neglected (at least at this early stage of research). We state that one connection transferring 100 bytes is better than 2 connections transferring 40 bytes (better means “less efforts”), so:

- 1/100 is less than 2/40 –> 0,01 is less than 0,05 –> which is true

So we can express the overall communication efforts of an algorithm as:

- C = n / S

After transforming the above energy related equations and disregarding hardware-specific values and constants, the main performance numbers of interest can be derived. The two key-numbers expressing the efficiency of the mobile system are:

1. **Mobile node number of consumed processor cycles: W**
2. **Mobile node communication efforts: C** (=n/S)

Gathering Results in Experiments:

To find out whether the desired objectives are met, the idea of the MECoS approach has to be compared to the traditional approach by utilising the measurements as described above. The measurements provide valuable significance for energy efficiency and resource consumption of the system. The most resource-intensive tasks of a mobile device are opening up a data connection, active data transfer and active processor (in decreasing order).

7.1.2 Overview of Results

The outcome of all experiments performed and all numbers that were measured can be depicted in a compressed way by comparing the calculated average overall system load (O= W*C) of the two approaches as shown in figure 7.3:

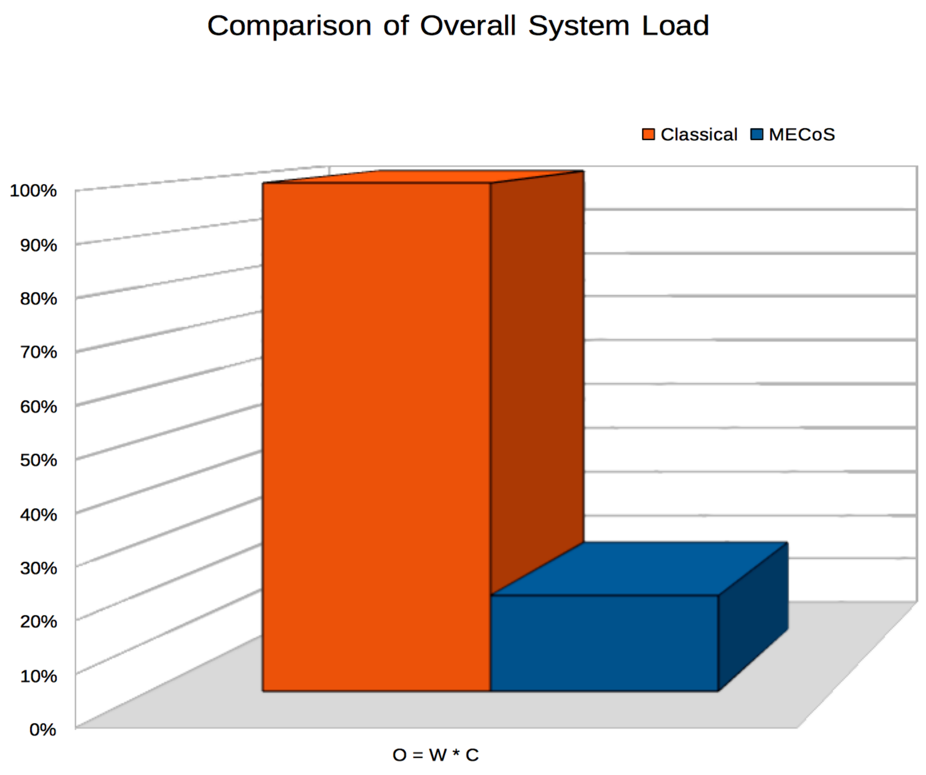


Figure 7.3: Overall System Load of MECoS compared to CS

Numbers that the above chart is based on (W - processor cycles and C - communication efforts) can be seen in table 7.1. It shows the overall sums of the two key numbers measured within the experiments, comparing the presented MECoS approach to a traditional client-server based approach (CS):

	Client Server	MECoS	**Improvement**
Proc. Cycles W[1]	180704097991047	126477271081977	**30%**
Comm. Efforts C [2]	0,000095882777159	0,000026639329203	**62%**

Table 7.1: Results Summary

A description of how the numbers W and C are calculated and which numbers they are based on can be found in the previous section (see 7.1.1). The bar chart and the summary table clearly show that *the presented MECoS approach outperforms traditional approaches in processing and communication efforts*.

7.2 Detailed Results Measured

Results have been measured within 95 different experiment runs, experiments were performed with different dataset widths and different databases to get a broader characteristic of how the two compared algorithms perform under different circumstances. For every experiment that was performed, the basic parameters and many characteristic numbers have been measured to analyse and point out resource consumption, efficiency and workload of the systems while they were active, for instance:

- dataset width
- database size / datasets processed (number of records)
- mode (MECoS or client server CS)
- processor cycles used
- average classification time per dataset on client
- average duration for a learning step of the evolving network on client
- number of data connections that were opened to the back-end system

- data connection open, active and idle periods
- number of bytes read and written by the client (overall bytes transferred between client and server)
- number of evolving nodes that existed in the evolving network structure on the client at the end of the experiment (classifier structure complexity)
- etc.

The detailed table of numbers measured throughout the experiments can be found in the appendix of this thesis (A).

All of them are quite interesting in some way but to highlight the achievement of the aims of this research, the collected measurements were selected and compressed into the three most important measurements that point out the efficiency of the algorithm (which are the basis of the calculated processor cycles C and communication efforts W numbers):

- overall number of processor cycles used
- overall data transfer between client and back-end node
- number of data connections opened

These three performance numbers are the most important ones regarding energy consumption and workload on the mobile device.

7.2.1 Processor Cycles

As shown in figure 7.4 the MECoS approach performs much better in terms of used processor cycles on the client than the Client Server (CS) approach. Only in a single experiment, were the used processor cycles nearly equal, in all others CS was outperformed by MECoS at about an average of 70% of CS.

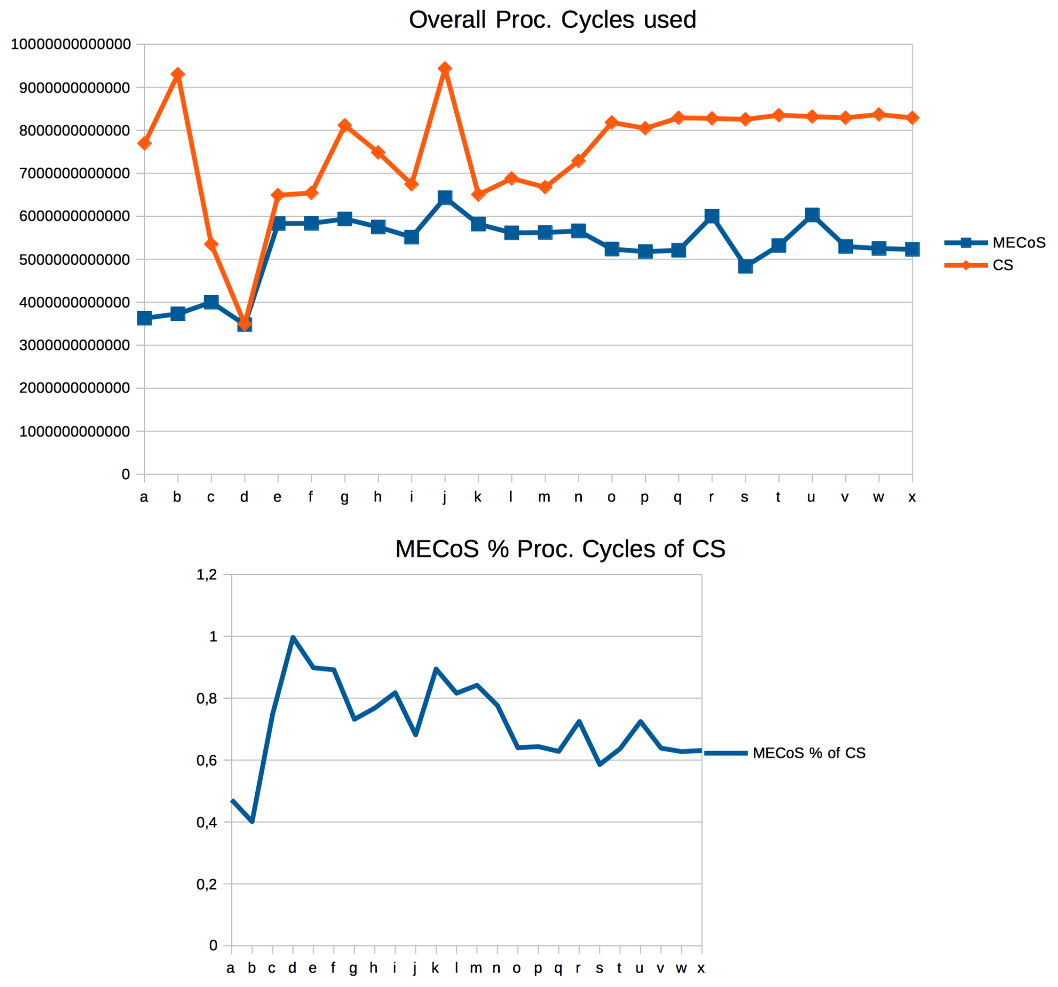


Figure 7.4: Overall Proc. Cycles used

7.2.2 Average Classification Time

Very similar to the processor cycles used as explained above, figure 7.5 below shows that the MECoS approach also performs better in terms of average classification time per dataset on the client. Classification time clearly depends on the processor and is tightly related to the number of processor cycles used, the close relation is also visible when comparing the curves of average classification time to the ones of used processor cycles. This does not bring about new facts but can be seen as validation of the measurements (if the curves were not related, this would be a strong indication for an error in the experiments or measurements).

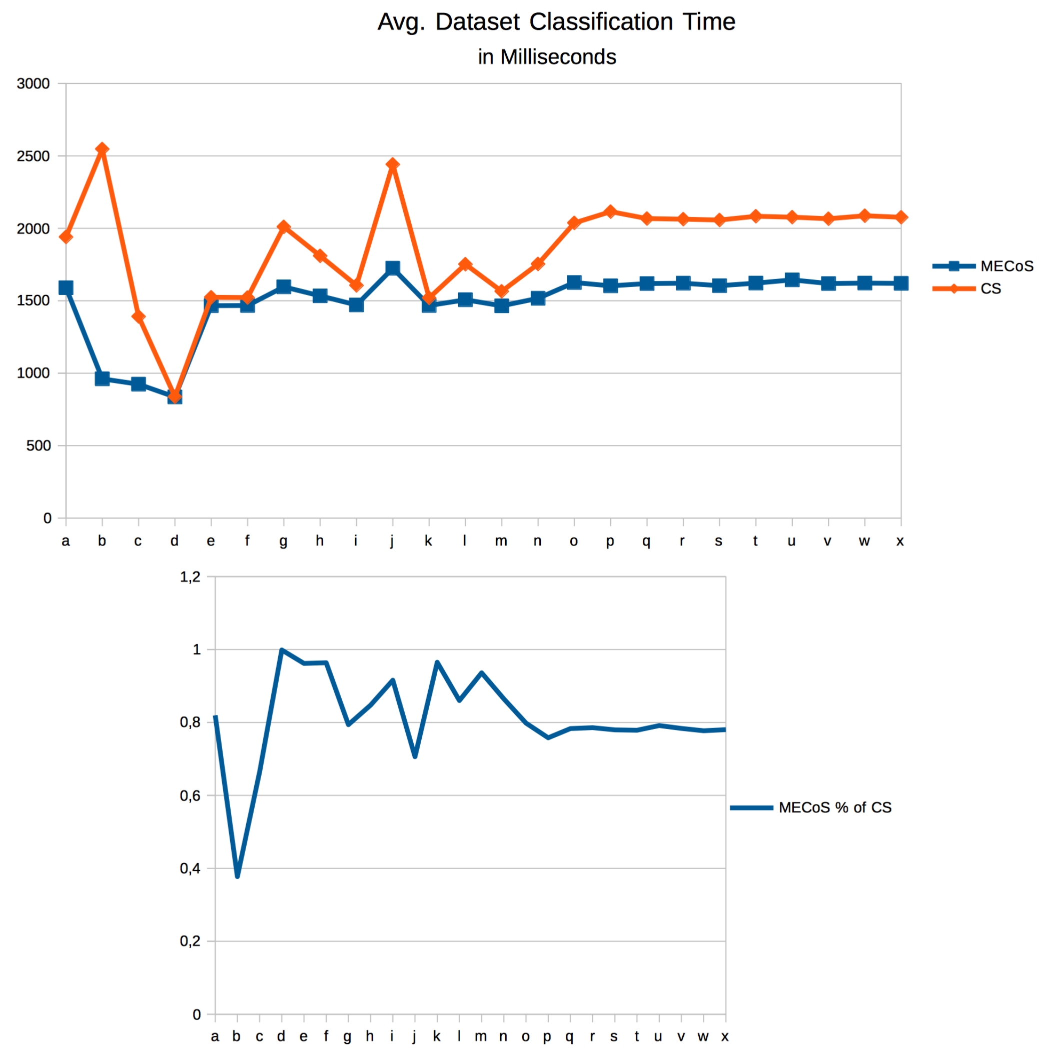


Figure 7.5: Avg. Dataset Classification Time

7.2.3 Overall Data Transfer

Figure 7.6 shows a possible drawback of the MECoS approach: the higher amount of data that has to be transferred (the moving process itself has to transfer the single knowledge base between the nodes). It can be seen that in ideal cases the required data transfer is only some percent higher than with the traditional CS approach, but on average the amount of data transfered is 150-170% higher with the presented MECoS approach than with CS.

It has to be mentioned that the MECoS implementation that was used within the experiments only applied a very simple RLE-based compression algorithm [172, 18, 35, 117]. It can be assumed that a refined version of the MECoS algorithm applying a better compression algorithm like LZW - Lempel-Ziv-Welsh ([152]) can drastically reduce the amount of data to be transferred and drop down the rate by which MECoS exceeds CS in this discipline.

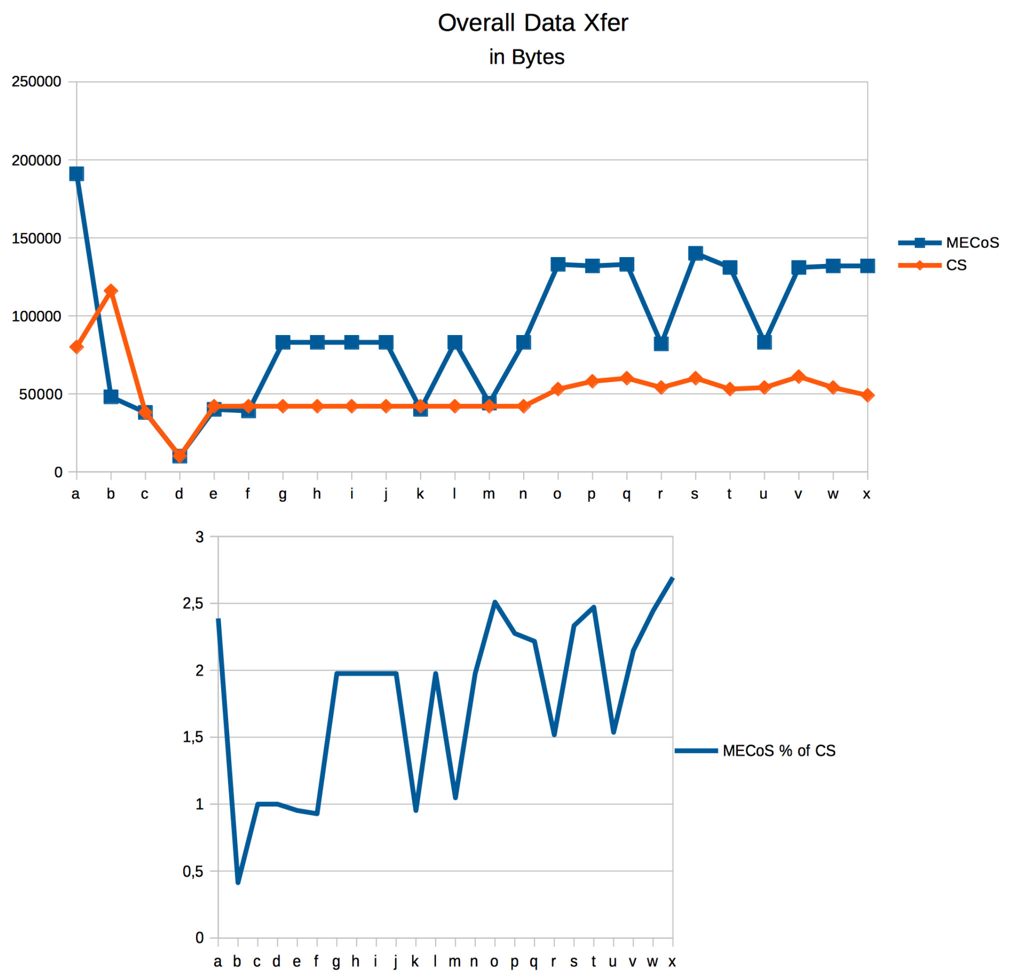


Figure 7.6: Overall Data Transfer

7.2.4 Connections Opened

The measured number of connections opened as seen in figure 7.7 shows that the communication efforts between client and back-end node are higher for the CS approach than for MECoS. The problem is that the CS approach cannot “delegate control” to back-end for an undefined amount of time (as MECoS does), but has to re-initiate communication in every potential emergency case. There CS is again outperformed by the MECoS approach. As the energy consumption for opening up a data connection is very high, a lower count is a great advantage. This also mitigates the drawback of the higher data transfer amount of MECoS approach as mentioned in the previous section.

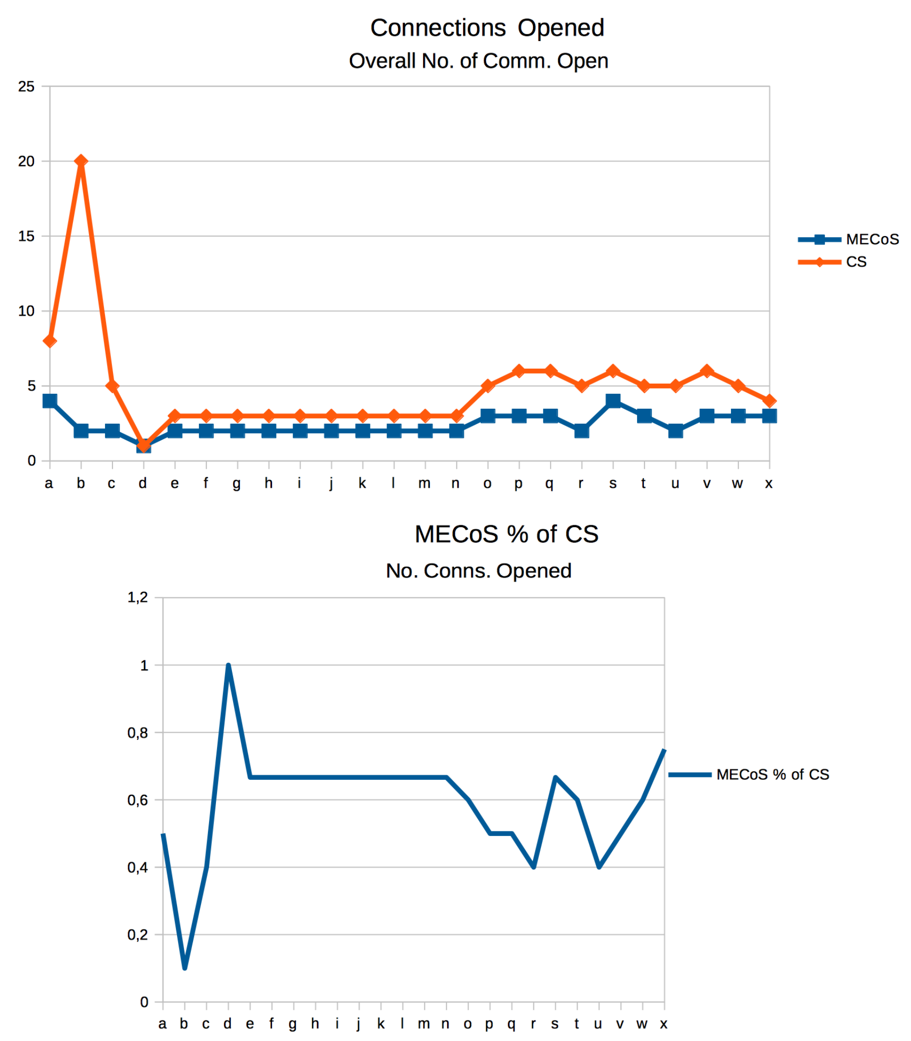


Figure 7.7: Overall Connections Opened

7.3 Discussion of Results

A significant improvement by the early prototype of the MECoS approach could be shown immediately. It runs more efficiently than traditional systems and at the same time provides an equivalent learning performance and classification quality.

The required amount of processor cycles on the mobile device is remarkably below traditional, CS based approaches and lies at about 70% on average, i.e. allows savings of processor load on the mobile device of about 30%. It is interesting that in contrast to these savings, the MECoS approach produces a higher amount of data transfer between the mobile and the back-end system, experiments showed that the overall amount of data transfer is at up to 150% compared to traditional approaches.

On the other hand, the MECoS prototype clearly outperformed the traditional approaches when taking into account the number of data connections that were opened between mobile and back-end device. Here the MECoS approach allows savings of about 40 percent. It is important to take this into account when looking at the higher quantity of overall data transferred at 150%, because energy consumption characteristics determine the significance of these numbers and their impact on the overall results:

- Processor: energy intensive, but rather low when compared to communication efforts (GPRS data link)
- Data transmission: actively sending/receiving data: energy intensive, drains battery

- Opening up a data connection: most energy-intensive. When GPRS/UMTS data links are opened, so-called "bursts" are required to establish the connection ([208, **?**]). These are short but extremely energy-intensive radio peaks of the cellphone or GSM modem. So, the impact of a higher amount of overall data transfer can be dropped by the remarkably lower number of data connections that have to be opened.

Remark on measured Communication Efforts

The MECoS prototype compressed the data for state/knowledge transfer with a simple RLE [172, 18, 35, 117] algorithm only. So, data transferred by the MECoS approach can be reduced massively by the application of a better compression algorithm. It demonstrates the improvement of the MECoS approach over traditional approaches also in terms of communication efforts: the outcome of the experiments has already shown that MECoS requires less communication efforts than traditional approaches in terms of the number of connections between mobile and backend device. When more sophisticated compression algorithms (e.g. LZW [152] or similar) are integrated into the MECoS approach, the required communication efforts can be reduced even more significantly. Even if the better compression uses more CPU cycles, CPU is cheaper than GPRS/UMTS - so this is a better deal in the end.

Final Verdict

If all three key energy consuming tasks are considered with their weights in combination, the advantage of the MECoS approach is clearly visible and it is proven that the previously set aims could be achieved - to recapitulate the stated aim of this research:

"The aim of this research project is to discover a novel approach based on Artificial Intelligence (AI) and Machine Learning (ML) methods, that results in a high quality and maximum energy- and computationally efficient operating Mobile Care System. This should enable a very economical, long period of autonomous operation without sacrificing the systems classification abilities / intelligence and result in applicable and cost-efficient mobile patient care systems."

As the results of experiments show, the presented MECoS algorithm provides the same classification quality as other state-of-the-art AI-based monitoring systems by working much more resource and energy efficiently and therefore provides a significantly longer period of autonomous operation of the system - the research aim is clearly met.

Even though the development and advancement of mobile hardware is permanently progressing and devices work more efficiently and additionally also battery capacities are permanently increasing, the requirement remains - for a very efficient mode of operation of Mobile Care Systems for achieving a maximum autonomous system runtime - and derived from this - achieve cost savings. Faster processors and data transfer rates have an impact on numbers but the proportional superiority of the MECoS approach in relation to traditional client-server based approaches will stay more or less unchanged: Mobile Care Systems that are based on the MECoS concept will always outperform traditional, client-server based systems because they inherently work more efficiently and therefore will always result in a longer autonomous system runtime.

A fundamental change in the results of this investigation would only occur if the proportions of the energy intensities for the performed tasks changed, i.e. required energy for processor, data communication and opening up data connections - in this case the results would have to be re-evaluated. But from the current point of view, this characteristic change is not to be expected in the near future.

Both measured systems, MECoS and also CS approaches, were based on simulation: in both cases the input data was generated by a developed software algorithm (both algorithms were fed with identical input data) and the classifiers were characteristic prototypes. To gather more detailed results, further experiments would be required, based on physiological input data (sensors measuring data from real patients) that are being measured in real world scenarios. This of course could result in (slight) changes of the outcome, but exceed the frame of this research project and it is very unlikely that the positive test of the stated hypothesis will be reverted.

Chapter 8

Conclusion

8.1 Original Research Questions Addressed

As presented in the introduction of this thesis (see Chapter 1), the original research question that set out this thesis years ago is:

"How can we design a Mobile Care System that balances the conflicting needs of providing real-time complex distributed intelligence, using limited resources in terms of computational power and power requirements?"

The aim of this research project is to discover a novel approach based on Artificial Intelligence (AI) and Machine Learning (ML) methods, that results in a high quality and maximum energy- and computationally efficiently operating Mobile Care System. This should enable a very economical, long period of autonomous operation without sacrificing the systems classification abilities / intelligence and result in applicable and cost-efficient mobile patient care systems. Broader and increased usage of these systems will be the consequence.

After a detailed investigation of the state of the art in patient monitoring and artificial intelligence systems, a novel approach could be presented that addresses the above research questions, the so-called MECoS approach: The basic idea of the MECoS approach was to utilise modern wireless networks to overcome the limited computational resources of the mobile monitoring devices through integration of the assistance of powerful back-end systems with (nearly) unlimited power. The idea was, that the back-end systems should take over control in extraordinary cases - this should allow complex tasks and high classification quality by allowing the mobile monitoring devices to stay rather simple, portable and provide a very long autonomous system runtime. Technically, a heterogeneous distributed system was built, in which a classical online machine learning algorithm was deployed. The speciality in this case is that the online machine learning algorithm works in a distributed way, i.e. the machine learning algorithm moves over multiple nodes within the system and integrates different available input parameters advancing a unique shared knowledge base on all nodes - so things learned on the back-end system are immediately available on the limited mobile monitoring device which would never have been able to learn these things itself.

A drawback of the MECoS approach is the rather high communicational efforts, as the knowledge base has to be transferred between the devices. It was unclear if this drawback would scatter the advantages of the MECoS approach, but as experiments have shown, data compression algorithms can be integrated and in the end the MECoS approach still offers significantly better results than the compared classical client server based approaches do.

8.2 Conclusion

The novel MECoS approach offers a significant advance in the application of a distributed machine learning algorithm that combines a battery-powered device with limited computational power to a remote and more powerful machine:

- Although running on a limited and portable device, through the distributed online machine learning algorithm the proposed system achieves exactly the same classification quality as a powerful stationary computer system can provide.

- The overall processor cycles (i.e. energy usage and resource consumption) used on the mobile device per test run, shows MECoS clearly in front of classical CS (client server) approaches: MECoS is mainly operating at about 70% of CS' consumption. Also at overall efforts for communication tasks MECoS outperforms CS with about 38% of CS' consumption.

The presented MECoS approach represents a distributed online learning process that operates highly efficiently within an unbalanced distributed environment. Because there is only a single, shared knowledge base in combination with a moving classifier instance, a very dynamic and energy-efficient operation of the system can be achieved. The classifier provides high-quality results without the drawbacks of current Mobile Care Systems ("semi-portable" vs. "semi-intelligent" as mentioned above, see section 2.4).

8.3 Future Direction

Even though it is only a first prototype, the MECoS System implementation already outperforms other approaches as the results have shown. Some aspects, however, were left open and several issues are subject to further optimisations. The direction for further research and improvement of the MECoS System concept should focus on:

Reducing Data Transfer The rather high amount of MECoS' data transfer has to be reduced, mainly by evaluation of best-fitting compression algorithms for representing the knowledge base (the serialised evolving net structure) and its transfer between client and back-end node and vice-versa (use other compression algorithms, chunked transfer, caching, etc.).

Classifier Parameter Tuning Another point that has to be investigated is how the system works with "real world" data, as current experiments were only performed with generated data. It can be expected that the classifiers parameters will have to be tuned to provide more qualitative results with real world data, as the current implementation is tuned for the generated data. It will be inevitable that experiments with real human patients will need to be performed at a later stage of this field of research (e.g. if a product based on the MECoS concept is to be developed).

Investigate Dynamic Sampling Frequency and Degree of Abnormality Investigation of energy aware computing (see 3.2.3) presented the concept of dynamic sampling frequency and the so-called degree of abnormality (which is a number that controls the dynamic sampling). The prototype implementation of the MECoS system was based on a multilayer perceptron and the outcome of the classifier was interpreted as no-alarm or alarm-classes. The concept of the degree of abnormality has not (yet) been implemented as this basically could be implemented for all Mobile Care Systems classifiers and is not a benefit limited to MECoS based systems only. Nevertheless it would be interesting to implement this concept both in the MECoS prototype and also in the classical Client Server based approach and investigate how this would influence the results.

Memory Footprint As portable devices only have very scarce resources, the memory footprint of the mobile classifier (client node) should be investigated and optimised as well.

Integration of future trends on ECoS based research In the meantime, research on ECoS-based systems has made progress. Over the last years, new variants of ECoS based systems have been presented. Prof. Kasabov has extended the ECoS paradigm from classical ANNs to spiking ANNs as presented in [106, 107, 105, 99, 98, 100]for instance. The impact of these recent updates on ECoS based systems has to be investigated and the MECoS approach can probably be enhanced further.

Comparison with alternative evolving classifiers Evaluation of popular AI and ML techniques as presented in chapter 4 showed that basically several approaches are available that could possibly be used for mobile patient monitoring tasks. I have selected the ECoS paradigm as a foundation for my further research because compared to the other techniques, it seems to be the best-fitting approach for addressing the research aims of this thesis to me. It would be very interesting to build similar systems with a “moving classifier” but based on alternative AI concepts like Decision Trees/C4.5 algorithm (see 4.3.5), Artificial Immune Systems (see 4.3.9), Multi-Objective Systems (see 4.3.8) or other Dynamic/Evolving Neural Network approaches (see 4.3.10) and compare them with the presented MECoS approach.

Bibliography

[1] http://www.tunstall.co.uk/what-we-do/telehealth. Tunstall UK, Telecare Products. Last accessed 18.09.2014.

[2] http://www.medvivo.com/our-solutions/telecare/. Medvivo UK - Telecare and Telehealth Solutions. Last accessed 18.09.2014.

[3] http://www.qualcomm.com/media/releases/2011/09/07/qualcomm-and-life-care-networks-launch-3g-mobile-health-project-help. Qualcomm Inc., 3G Mobile Health Project. Last accessed 18.09.2014.

[4] http://www.radio-electronics.com/info/cellulartelecomms/gsm_technical/gsm-radio-air-interface-slot-burst.php/. GSM Radio Air Interface, GSM Slot & Burst. Last accessed 18.09.2014.

[5] http://archive.ics.uci.edu/ml/. UCI Machine Learning Repository. Last accessed 18.09.2014.

[6] http://www.oracle.com/technetwork/java/index.html. Java Technology, Oracle/Sun. Last accessed 18.09.2014.

[7] `http://www.oracle.com/technetwork/java/javame/javamobile/download/overview/index.html`. Java Wireless Toolkit/Java ME SDK, Oracle/Sun. Last accessed 18.09.2014.

[8] `https://github.com/jimmikaelkael/multi-layer-perceptron`. Multilayer Perceptron implemented in Java. Last accessed 18.09.2014.

[9] `https://code.google.com/p/scalalab/wiki/JavaNeuralNetworkFrameworkExamplesInJava`. Java Neural Network Framework (neuroph) Examples. Last accessed 18.09.2014.

[10] `http://www.winterwell.com/software/compressor.php`. Java/J2ME implementation of a Range Encoding compression. Last accessed 18.09.2014.

[11] `http://www.dclausen.net/projects/microfloat/`. MicroFloat Java Software Library. Last accessed 18.09.2014.

[12] Open at bluetooth application note, revision 002 december 2005. (see appendix D.1).

[13] Open at power consumption modes application note revision 004, september 2005. (see appendix D.2).

[14] Wavecom wismo quik q2406 and q2426 product specification revision 005, february 2005. (see appendix D.3).

[15] Wismo quik q2686 product technical specification revision 005, february 2006. (see appendix D.4).

[16] Shigeo Abe. *Support vector machines for pattern classification*. Springer, 2010.

[17] Natalia Adrienko and Gennady Adrienko. Spatial generalization and aggregation of massive movement data. *Visualization and Computer Graphics, IEEE Transactions on*, 17(2):205–219, 2011.

[18] S Akhter and MA Haque. Ecg compression using run length encoding. *European signal processing (EUSIPCO-2010)*, pages 1645–1649, 2010.

[19] Ammar Almomani, BB Gupta, Tat-chee Wan, Altyeb Altaher, and Selvakumar Manickam. Phishing dynamic evolving neural fuzzy framework for online detection zero-day phishing email. *arXiv preprint arXiv:1302.0629*, 2013.

[20] J. Ammerlaan and D. Wright. Adaptive cooperative fuzzy logic controller. *27th Australasian Computer Science Conference (ACSC2004)*, pages 255–263, 2004.

[21] Michele Amoretti, Sergio Copelli, Folker Wientapper, Francesco Furfari, Stefano Lenzi, and Stefano Chessa. Sensor data fusion for activity monitoring in the persona ambient assisted living project. *Journal of Ambient Intelligence and Humanized Computing*, 4(1):67–84, 2013.

[22] G Andia Vera, Apostolos Georgiadis, Ana Collado, and Selva Via. Design of a 2.45 ghz rectenna for electromagnetic (em) energy scavenging. In *Radio and Wireless Symposium (RWS), 2010 IEEE*, pages 61–64. IEEE, 2010.

[23] K.K. Ang and C. Quek. Stock trading using rspop: A novel rough set-based neuro-fuzzy approach. *IEEE Transactions on Neural Networks*, 17(5):1301–1315, 2006.

[24] Senem Kursun Bahadir, Selcuk Cebi, Cengiz Kahraman, and Fatma Kalaoglu. Developing a smart clothing system for blinds based on information axiom. *International Journal of Computational Intelligence Systems*, 6(2):279–292, 2013.

[25] Roman M Balabin and Sergey V Smirnov. Variable selection in near-infrared spectroscopy: benchmarking of feature selection methods on biodiesel data. *Analytica chimica acta*, 692(1):63–72, 2011.

[26] Rahul Balani. Energy consumption analysis for bluetooth, wifi and cellular networks. *Online]. http://nesl. ee. ucla. edu/fw/documents/reports/2007/PowerAnalysis. pdf*, 2007.

[27] Regis J Bates. *GPRS: general packet radio service*. McGraw-Hill Professional, 2001.

[28] Sophie Beale, Diana Sanderson, and Jen Kruger. Evaluation of the telecare development programme. *York Health Economics Consortium*, 2009. Published by the Scottish Government, January 2009.

[29] Lubica Benuskova and Nikola K Kasabov. *Computational neurogenetic modeling*. Springer, 2010.

[30] D.P. Bertsekas and J.N. Tsitsiklis. Neuro-dynamic programming: an overview. *Proceedings of the 34th IEEE Conference on Decision and Control*, 1:560–564, December 1995.

[31] Mr Ketan D Bodhe, RR Sawant, and Mr AN Kazi. A proposed mobile based health care system for patient diagnosis using android os. 2014.

[32] Matthias Borowski, Sylvia Siebig, Christian Wrede, and Michael Imhoff. Reducing false alarms of intensive care online-monitoring systems: an evaluation of two signal extraction algorithms. *Computational and mathematical methods in medicine*, 2011, 2011.

[33] Oliver J Bott, Ina Hoffmann, Joachim Bergmann, Nathalie Gusew, Oliver Schnell, Enrique J Gomez, M Elena Hernando, Patrick Kosche, Christian von Ahn, Dirk C Mattfeld, et al. His modelling and simulation based cost–benefit analysis of a telemedical system for closed-loop diabetes therapy. *international journal of medical informatics*, 76:S447–S455, 2007.

[34] David G Bounds, Paul J Lloyd, Bruce Mathew, and Gordon Waddell. A multilayer perceptron network for the diagnosis of low back pain. In *Neural Networks, 1988., IEEE International Conference on*, pages 481–489. IEEE, 1988.

[35] Stevan D Bradley. Optimizing a scheme for run length encoding. *Proceedings of the IEEE*, 57(1):108–109, 1969.

[36] M. Braecklein, I Tchoudovski, C Moor, M. Werthmann, S. Carson, and A. Bolz. Drahtlose kardiolog. Überwachung von risikopatienten in häuslicher umgebung. *Health Academy*, (01/2004):188–195, 2004.

[37] J. Cabestany, J.M. Moreno, C. Perez, A. Sama, A. Catala, A. Rodriguez-Molinero, and M. Arnal. Fate: One step towards an automatic aging people fall detection service. In *Mixed Design of Integrated Circuits and Systems (MIXDES), 2013 Proceedings of the 20th International Conference*, pages 545–552, June 2013.

[38] Aaron Carroll and Gernot Heiser. An analysis of power consumption in a smartphone. In *USENIX annual technical conference*, pages 271–285, 2010.

[39] Jiyoung Chang, Michael Dommer, Chieh Chang, and Liwei Lin. Piezoelectric nanofibers for energy scavenging applications. *Nano Energy*, 1(3):356–371, 2012.

[40] Min Chen, Sergio Gonzalez, Athanasios Vasilakos, Huasong Cao, and Victor C Leung. Body area networks: A survey. *Mobile Networks and Applications*, 16(2):171–193, 2011.

[41] Jin Young Choi and Chong-Ho Choi. Sensitivity analysis of multilayer perceptron with differentiable activation functions. *Neural Networks, IEEE Transactions on*, 3(1):101–107, 1992.

[42] Patrick Marques Ciarelli, Elias Oliveira, and Evandro O.T. Salles. An incremental neural network with a reduced architecture. *Neural Networks*, 35:70–81, 2012.

[43] Ferdinando Cicalese, Antonia di Nola, and Vincenzo Loia. A fuzzy evolutionary framework for adaptive agents. *SAC '99*, pages 233–237, 1999. ACM.

[44] Marleen C Cloostermans, Cecile C de Vos, and Michel JAM van Putten. A novel approach for computer assisted eeg monitoring in the adult icu. *Clinical neurophysiology*, 122(10):2100–2109, 2011.

[45] Ângelo Costa, José Carlos Castillo, Paulo Novais, Antonio Fernández-Caballero, and Ricardo Simoes. Sensor-driven agenda for intelligent home care of the elderly. *Expert Systems with Applications*, 39(15):12192–12204, 2012.

[46] Richard Cracknell. The ageing population. *Key Issues for the new parliament 2010*, pages 44–45, 2007. House of Commons Library Research.

[47] Qingguang Cui, Matthew O Ward, Elke A Rundensteiner, and Jing Yang. Measuring data abstraction quality in multiresolution visualizations. *Visualization and Computer Graphics, IEEE Transactions on*, 12(5):709–716, 2006.

[48] LA Curtis. *Unit costs of health and social care 2012*. Personal Social Services Research Unit, 2012.

[49] Maria Cvach. Monitor alarm fatigue: an integrative review. *Biomedical Instrumentation & Technology*, 46(4):268–277, 2012.

[50] Dipankar Dasgupta. Artificial neural networks and artificial immune systems: similarities and differences. In *Systems,*

Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on, volume 1, pages 873–878. IEEE, 1997.

[51] Dipankar Dasgupta, Senhua Yu, and Fernando Nino. Recent advances in artificial immune systems: Models and applications. *Applied Soft Computing*, 11:1574–1587, 2011.

[52] Omid E David and Iddo Greental. Genetic algorithms for evolving deep neural networks. In *Proceedings of the 2014 conference companion on Genetic and evolutionary computation companion*, pages 1451–1452. ACM, 2014.

[53] Leandro Nunes De Castro and Jonathan Timmis. *Artificial immune systems: a new computational intelligence approach*. Springer, 2002.

[54] Artem Dementyev, Steve Hodges, Stuart Taylor, and Joshua Smith. Power consumption analysis of bluetooth low energy, zigbee and ant sensor nodes in a cyclic sleep scenario. In *Wireless Symposium (IWS), 2013 IEEE International*, pages 1–4. IEEE, 2013.

[55] CM Denkinger, J Grenier, AK Stratis, A Akkihal, N Pant-Pai, and M Pai. Mobile health to improve tuberculosis care and control: a call worth making [review article]. *The International Journal of Tuberculosis and Lung Disease*, 17(6):719–727, 2013.

[56] George Dimitoglou, James A Adams, and Carol M Jim. Comparison of the c4. 5 and a naive bayes classifier for the prediction of lung cancer survivability. *arXiv preprint arXiv:1206.1121*, 2012.

[57] Thomas G. Ditterich. Machine learning for sequential data: A review. *Lecture Notes in Computer Science*, 2396:15–30, 2002. Springer Verlag.

[58] George Dounias and Derek Linkens. Adaptive systems and hybrid computational intelligence in medicine. *Artificial Intelligence in Medicine*, 32(2004 (32)):151–155, 2004. Elsevier.

[59] Andries P Engelbrecht. *Fundamentals of computational swarm intelligence*. John Wiley & Sons, 2006.

[60] Orhan Er, Nejat Yumusak, and Feyzullah Temurtas. Diagnosis of chest diseases using artificial immune system. *Expert Systems with Applications*, 39:1862–1868, 2012.

[61] Richard L Ernst and Joel W Hay. The us economic and social costs of alzheimer's disease revisited. *American Journal of Public Health*, 84(8):1261–1264, 1994.

[62] Jonny Farringdon, Andrew J Moore, Nancy Tilbury, James Church, and Pieter D Biemond. Wearable sensor badge and sensor jacket for context awareness. In *Wearable Computers, 1999. Digest of Papers. The Third International Symposium on*, pages 107–113. IEEE, 1999.

[63] Michela Fazzolari, Rafael Alcala, Yusuke Nojima, Hisao Ishibuchi, and Francisco Herrera. A review of the application of multiobjective evolutionary fuzzy systems: Current status and further directions. *Fuzzy Systems, IEEE Transactions on*, 21(1):45–65, 2013.

[64] Laura Marie Feeney and Martin Nilsson. Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In *INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 3, pages 1548–1557. IEEE, 2001.

[65] MVM Figueredo and JS Dias. Mobile telemedicine system for home care and patient monitoring. In *Engineering in Medicine and Biology Society, 2004. IEMBS'04. 26th Annual International Conference of the IEEE*, volume 2, pages 3387–3390. IEEE, 2004.

[66] J. Flinn, S.Y. Park, and M. Satyanarayanan. Balancing performance, energy, and quality in pervasive computing. In *Proc. o.t. 22nd Int. Conf. on Dist. Comp. Systems*, pages 217–226. ICDCS 02, IEEE Society, 2002.

[67] MW Gardner and SR Dorling. Artificial neural networks (the multilayer perceptron)–a review of applications in the atmospheric sciences. *Atmospheric environment*, 32(14-15):2627–2636, 1998.

[68] Antoniya Georgieva, Stephen J Payne, Mary Moulden, and Christopher WG Redman. Artificial neural networks applied to fetal monitoring in labour. *Neural Computing and Applications*, 22(1):85–93, 2013.

[69] Wulfram Gerstner and Werner M Kistler. *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[70] A. Ghobakhlou, M. Watts, and N. Kasabov. Adaptive speech recognition with evolving connectionist systems. *Elsevier Information Sciences*, (156):71–83, 2003.

[71] L. Goh, Q. Song, and N. Kasabov. A novel feature selection method to improve classifcation of gene expression data. *2nd Asia-Pacific Bioinformatics Conference (APBC2004)*, pages 161–166, 2004.

[72] Faustino Gomez and Risto Miikkulainen. Incremental evolution of complex general behavior. *Adaptive Behavior*, 5(3-4):317–342, 1997.

[73] Kelly Creighton Graham and Maria Cvach. Monitor alarm fatigue: standardizing use of physiological monitors and decreasing nuisance alarms. *American Journal of Critical Care*, 19(1):28–34, 2010.

[74] John J Grefenstette. *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*. Psychology Press, 2013.

[75] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *The Journal of Machine Learning Research*, 3:1157–1182, 2003.

[76] H. Hagras, V. Callaghan, M. Colley, and G. Clarke. A hierarchical fuzzy-genetic multi-agent architecture for intelligent buildings online learning, adaption and control. *Elsevier Information Sciences*, (150):33–56, 2003.

[77] Timo Halonen, Javier Romero, and Juan Melero. *GSM, GPRS and EDGE performance: evolution towards 3G/UMTS*. John Wiley & Sons, 2004.

[78] Neil A Halpern and Stephen M Pastores. Critical care medicine in the united states 2000–2005: An analysis of bed numbers, occupancy rates, payer mix, and costs*. *Critical care medicine*, 38(1):65–71, 2010.

[79] Matthew Hausknecht, Joel Lehman, Risto Miikkulainen, and Peter Stone. A neuroevolution approach to general atari game playing. 2013.

[80] Jian He, Chen Hu, and Yang Li. An autonomous fall detection and alerting system based on mobile and ubiquitous computing. In *Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC)*, pages 539–543. IEEE, 2013.

[81] Martia A. Hearst. Support vector machines. *IEEE Intelligent Systems*, (July/August 1998):18–28, July 1998.

[82] Friedhelm Hillebrand. *GSM and UMTS: the creation of global mobile communication*. John Wiley & Sons, Inc., 2002.

[83] Shashivadan Parbat Hirani, Michelle Beynon, Martin Cartwright, Lorna Rixon, Helen Doll, Catherine Henderson, Martin Bardsley, Adam Steventon, Martin Knapp, Anne Rogers, Peter Bower,

Caroline Sanders, Ray Fitzpatrick, Jane Hendy, and Stanton Peter Newman. The effect of telecare on the quality of life and psychological well-being of elderly recipients of social care over a 12-month period: the whole systems demonstrator cluster randomised trial. *Age and Ageing*, 2013.

[84] Steven A. Hofmeyr and Stephanie A. Forrest. Architecture for an artificial immune system. *Evol. Comput.*, 8(4):443–473, December 2000.

[85] G.B. Huang, N.Y. Liang, H.J. Rong, P. Saratchandran, and N. Sundararajan. On-line sequential extreme learning machine. *the IASTED International Conference on Computational Intelligence (CI 2005), Calgary, Canada, July 4-6, 2005*, 2005.

[86] John E. Hunt and Denise E. Cooke. Learning using an artificial immune system. *Journal of Network and Computer Applications*, 19(2):189 – 212, 1996.

[87] I. Hunter. Telecare: the future? *Ferryhill Gazette*, 2010.

[88] Michael Imhoff and Silvia Kuhls. Alarm algorithms in critical care monitoring. *Anesthesia & Analgesia*, 102(5):1525–1537, 2006.

[89] R Istepanian, N Philip, XH Wang, and S Laxminarayan. Non-telephone healthcare: The role of 4g and emerging mobile systems for future m-health systems. In *Future Visions on Biomedicine and Bioinformatics 2*, pages 9–16. Springer, 2011.

[90] Philip Jacobs and Thomas W Noseworthy. National estimates of intensive care utilization and costs: Canada and the united states. *Critical care medicine*, 18(11):1282–1286, 1990.

[91] C Jean-Mistral, S Basrour, and JJ Chaillout. Modelling of dielectric polymers for energy scavenging applications. *Smart Materials and Structures*, 19(10):105006, 2010.

[92] Anthony F Jerant, Rahman Azari, and Thomas S Nesbitt. Reducing the cost of frequent hospital admissions for congestive heart failure: a randomized trial of a home telecare intervention. *Medical care*, 39(11):1234–1245, 2001.

[93] N. Kannathal, C.M. Lim, U. Rajendra Acharya, and P.K. Sadasivan. Cardiac state diagnosis using adaptive neuro-fuzzy technique. *Medical Engineering & Physics*, 28:809–815, 2006.

[94] Oğuz Karan, Canan Bayraktar, Haluk Gümüşkaya, and Bekir Karlık. Diagnosing diabetes using neural networks on small mobile devices. *Expert Systems with Applications*, 39(1):54–60, 2012.

[95] N. Kasabov. Ecos: Evolving connectionist systems and the eco learning paradigm. In *Proceedings of ICONIP'98, Kitakyushu*. Department of Information Science, University of Otago, October 1998.

[96] N. Kasabov. Evolving connectionist systems for on-line, knowledge-based learning: Principles and applications. *The Information Science Discussion Paper Series*, (99/02), March 1999.

[97] N Kasabov. Mapping, learning and mining of spatiotemporal brain data with 3d evolving spiking neurogenetic models. 2012.

[98] N Kasabov. Spatio-temporal pattern recognition with evolving spiking neural networks. 2012.

[99] N Kasabov. Neurocomputation as brain inspired informatics: methods, systems, applications. Panhellenic Conference on Informatics, 2013.

[100] N Kasabov. Neurocomputing for spatio-/spectro temporal pattern recognition and early event prediction: methods, systems, applications. 2013.

[101] N. Kasabov, M. Middlemiss, and T. Lane. A generic connectionist -based method for on-line feature selection and modelling with a case study of gene expression. *Knowledge Engineering and Discovery Research Institute*, 2003? School of Information Technology, Auckland University of Technology.

[102] N. Kasabov, E. Postma, and van den J. Herik. Avis: a connectionist-based framework for integrated auditory and visual information processing. *Elsevier Information Sciences*, 123:127–148, 2000.

[103] N. Kasabov, D. Zhang, and P.S. Pang. Incremental learning in autonomous systems: Evolving connectionist systems for on-line image and speech recognition. *2005 IEEE Workshop on Advanced Robotics and Its Social Impacts*, pages 120–125, 2005.

[104] Nikola Kasabov. Evolving intelligence in humans and machines: Integrative evolving connectionist systems approach. *Computational Intelligence Magazine, IEEE*, 3(3):23–37, 2008.

[105] Nikola Kasabov. The evolution of the evolving neuro-fuzzy systems: From expert systems to spiking-, neurogenetic-, and quantum inspired. In *On Fuzziness*, pages 271–280. Springer, 2013.

[106] Nikola Kasabov, Kshitij Dhoble, Nuttapod Nuntalid, and Giacomo Indiveri. Dynamic evolving spiking neural networks for on-line spatio-and spectro-temporal pattern recognition. *Neural Networks*, 41:188–201, 2013.

[107] Nikola Kasabov, Valery Feigin, Zeng-Guang Hou, Yixiong Chen, Linda Liang, Rita Krishnamurthi, Muhaini Othman, and Priya Parmar. Evolving spiking neural networks for personalised modelling, classification and prediction of spatio-temporal patterns with a case study on stroke. *Neurocomputing*, 134:269–279, 2014.

[108] Nikola K. Kasabov. Evolving fuzzy neural networks for supervised/unsupervised on-line, knowledge based learning. *IEEE Transactions of Systems, Man and Cybernetics*, 31(6), 2001.

[109] Nikola K. Kasabov. On-line learning, reasoning, rule extraction and aggregation in locally optimized evolving fuzzy neural networks. *Elsevier Neurocomputing*, 41:2545, 2001.

[110] Nikola K. Kasabov. *Evolving Connectionist Systems*. Number 978-1-84628-345-1. Springer, second edition edition, 2007.

[111] N.K. Kasabov and Q. Song. Denfis: dynamic evolving neural-fuzzy inference system and itsapplication for time-series prediction. *IEEE Transactions on Fuzzy Systems*, 10(2):144–154, April 2002.

[112] James Kennedy, James F Kennedy, and Russell C Eberhart. *Swarm intelligence*. Morgan Kaufmann, 2001.

[113] Bahador Khaleghi, Alaa Khamis, Fakhreddine O Karray, and Saiedeh N Razavi. Multisensor data fusion: A review of the state-of-the-art. *Information Fusion*, 14(1):28–44, 2013.

[114] Jungwon Kim and P.J. Bentley. Towards an artificial immune system for network intrusion detection: an investigation of dynamic clonal selection. In *Evolutionary Computation, 2002. CEC '02. Proceedings of the 2002 Congress on*, volume 2, pages 1015–1020, 2002.

[115] Younghoon Kim, Sangyub Kim, Dongmin Kang, Hwangsoo Park, Nammoon Kim, Sung Hyun Yang, and Youngok Kim. A simple falling recognition scheme for a human body by using mobile devices. In *Proceedings of the 1st International Conference on Advanced Information and Comuter Technology (AICT), Warsawa, Poland*, pages 14–17, 2013.

[116] Halife Kodaz, Seral Özsen, Ahmet Arslan, and Salih Günes. Medical application of information gain based artificial immune recognition system (airs): Diagnosis of thyroid disease. *Expert Systems with Applications*, 36:3086–3092, 2009.

[117] SR Kodituwakku and US Amarasinghe. Comparison of lossless data compression algorithms for text data. *Indian journal of computer science and engineering*, 1(4):416–425, 2010.

[118] Nate Kohl and Risto Miikkulainen. Evolving neural networks for strategic decision-making problems. *Neural Networks*, 22:326—337, 2009.

[119] Bonhyun Koo, Kyusuk Han, James J Jong Hyuk Park, and Taeshik Shon. Design and implementation of a wireless sensor network architecture using smart mobile devices. *Telecommunication Systems*, 52(4):2311–2320, 2013.

[120] D. Kottmann, R. Wittmann, and M. Posur. Delegating remote operation execution in a mobile computing environment. *ACM Mobile Networks and Applications*, (1 (1996)):387–397, 1996.

[121] Prajakta Kulkarni and Yusuf Ozturk. mphasis: Mobile patient healthcare and sensor information system. *Journal of Network and Computer Applications*, 34(1):402 – 417, 2011.

[122] Ashok Kumar. Studies on inculcating electronics in textile to develop wearable electronics. 2013.

[123] Duck Hee Lee, Ahmed Rabbi, Jaesoon Choi, and Reza Fazel-Rezai. Development of a mobile phone based e-health monitoring application. *Development*, 3(3), 2012.

[124] Jin-Shyan Lee, Yu-Wei Su, and Chung-Chou Shen. A comparative study of wireless protocols: Bluetooth, uwb, zigbee, and wi-fi. In

Industrial Electronics Society, 2007. IECON 2007. 33rd Annual Conference of the IEEE, pages 46–51. IEEE, 2007.

[125] Ren-Guey Lee, Chun-Chieh Hsiao, Kuei-Chien Chen, and Ming-Hsio Liu. An intelligent diabetes mobile care system with alert mechanism. *Biomedical Engineering: Applications, Basis and Communications*, 17(04):186–192, 2005.

[126] Ren-Guey Lee, Chien-Chih Lai, Shao-Shan Chiang, Hsin-Sheng Liu, Chun-Chang Chen, and Guan-Yu Hsieh. Design and implementation of a mobile-care system over wireless sensor network for home healthcare applications. In *Engineering in Medicine and Biology Society, 2006. EMBS'06. 28th Annual International Conference of the IEEE*, pages 6004–6007. IEEE, 2006.

[127] Joel Lehman and Risto Miikkulainen. Neuroevolution. *Scholarpedia*, 8(6):30977, 2013.

[128] Daniel Leite, Pyramo Costa, and Fernando Gomide. Evolving granular neural networks from fuzzy data streams. *Neural Networks*, 38:1–16, 2013.

[129] Zhonghua Li, Yunong Zhang, and Hong-Zhou Tan. Ia-ais: An improved adaptive artificial immune system applied to complex optimization problems. *Applied Soft Computing*, 11:4692–4700, 2011.

[130] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan. A fast and accurate online sequential learning algorithm for

feedforward neural networks. *IEEE Transactions on Neural Networks*, 17(6):1411–1423, 2006.

[131] Wen-Te Liu, Chien-Da Huang, Chun-Hua Wang, Kang-Yun Lee, Shu-Min Lin, and Han-Pin Kuo. A mobile telephone-based interactive self-care system improves asthma control. *European Respiratory Journal*, 37(2):310–317, 2011.

[132] Benny Lo and Guang-Zhong Yang. Key technical challenges and current implementations of body sensor networks. *IEEE Proceedings of the 2nd intl. WS on BSN's*, (April 2005):1–5, 2005.

[133] T. Lo, Benny P.L.and Surapa, R. King, and Yang G. Body sensor network - a wireless sensor platform f. pervasive healthcare monitoring. *Lecture Notes in Computer Science*, Vol 3468:77–80, 2005.

[134] Brent Longstaff, Sasank Reddy, and Deborah Estrin. Improving activity classification for health applications on mobile devices using active and semi-supervised learning. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference on-NO PERMISSIONS*, pages 1–7. IEEE, 2010.

[135] George F. Luger. *Artificial Intelligence - Structures and Strategies for Complex Problem Solving*. Addison Wesley, 5th edition, 2005.

[136] Ziyu Lv, Feng Xia, Guowei Wu, Lin Yao, and Zhikui Chen. icare: a mobile health monitoring system for the elderly. In *Proceedings*

of the 2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing, pages 699–705. IEEE Computer Society, 2010.

[137] K. Malhi, S.C. Mukhopadhyay, J. Schnepper, M. Haefke, and H. Ewald. A zigbee-based wearable physiological parameters monitoring system. *Sensors Journal, IEEE*, 12(3):423–430, March 2012.

[138] R Timothy Marler and Jasbir S Arora. Survey of multi-objective optimization methods for engineering. *Structural and multidisciplinary optimization*, 26(6):369–395, 2004.

[139] Thomas L Martin, Daniel P Siewiorek, Asim Smailagic, Matthew Bosworth, Matthew Ettus, and Jolin Warren. A case study of a system-level approach to power-aware computing. *ACM Transactions on Embedded Computing Systems (TECS)*, 2(3):255–276, 2003.

[140] Risto Miikkulainen. Evolving neural networks. In *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, pages 2441–2460. ACM, 2010.

[141] Julie Mills and Emily Knipe. Historic and projected mortality data from the period and cohort life tables, 2012-based, uk, 1981-2062. *Office for National Statistics Statistical Bulletin*, December 2012. Office For National Statistics www.ons.gov.uk.

[142] Julie Mills and Emily Knipe. Interim life tables, england and wales, 2010-2012. *Office for National Statistics Statistical Bulletin*, October 2013. Office For National Statistics www.ons.gov.uk.

[143] Michael Mitchell, Christopher Meyers, A-IA Wang, and Gary Tyson. Contextprovider: Context awareness for medical monitoring applications. In *Engineering in Medicine and Biology Society, EMBC, 2011 Annual International Conference of the IEEE*, pages 5244–5247. IEEE, 2011.

[144] Tom M. Mitchell. *Machine Learning*. Number 0-07-115467-1. McGraw-Hill, international edition, 1997.

[145] Tom M. Mitchell. Machine learning and data mining. *Communications of the ACM*, November 1999/Vol. 42, No. 11:30–36, 1999.

[146] P. J. Modi and W. Shen. Collaborative multiagent learning for class. tasks. *Agents*, (May-June 01), May-June 01 2001.

[147] David E Moriarty and Risto Miikkulainen. Forming neural networks through efficient and adaptive coevolution. *Evolutionary Computation*, 5(4):373–399, 1997.

[148] Jeanette L Morrison, Qian Cai, Nancy Davis, Yan Yan, Michael L Berbaum, Michael Ries, and Glen Solomon. Clinical and economic outcomes of the electronic intensive care unit: Results from two community hospitals*. *Critical care medicine*, 38(1):2–8, 2010.

[149] Muhammad Mubashir, Ling Shao, and Luke Seed. A survey on fall detection: Principles and approaches. *Neurocomputing*, 100:144–152, 2013.

[150] D.J. Musliner, J.A. Hendler, and A.K. Agrawala. The challenges of real-time ai. *U. Maryland Technical Report CS-TR-3290, UMIACS-TR-94-69*, 1994.

[151] M. Negnevitsky. *Artificial Intelligence - A Guide to Intelligent Systems*. Number 0-201-71159-1. Addison-Wesley Publishing Company, 2002.

[152] Mark R. Nelson. Lzw data compression. *Dr. Dobb's Journal*, 14(10):29–36, October 1989.

[153] Taher Nikham, Rasoul Azizipanah-Abarghooee, and Mohammad Rasoul Narimani. A new multi objective optimization approach based on tlbo for location of automatic voltage regulators in distribution systems. *Engineering Applications of Artificial Intelligence*, 25:1577–1588, 2005.

[154] Nils J. Nilsson. *Introduction to Machine Learning*. unpublished/draft, 1996. Stanford University, Dept. of Computer Science.

[155] R. Nowicki and A. Pokropinska. Neuro-fuzzy structures and their comparative analysis. In *Proceedings of the 4th inl. Symposium on Information and Communication Technologies WISICT'05*, 2005.

[156] Office for National Statistics (ONS). Age structure of united kingdom 1971 - 2085. `http://www.ons.gov.uk/ons/interactive/uk-population-pyramid---dvc1/index.html`. Interactive Graphic from: http://www.ons.gov.uk/ons/interactive/uk-population-pyramid—dvc1/index.html. Last accessed 18.09.2014.

[157] Errol Ozdalga, Ark Ozdalga, and Neera Ahuja. The smartphone in medicine: A review of current and potential use among physicians and students. *Journal of Medical Internet Research*, 14(5):e128, 2012.

[158] Mahesh Pal and Giles M Foody. Feature selection for classification of hyperspectral data by svm. *Geoscience and Remote Sensing, IEEE Transactions on*, 48(5):2297–2307, 2010.

[159] Sankar K Pal and Sushmita Mitra. Multilayer perceptron, fuzzy sets, and classification. *IEEE Transactions on Neural Networks*, 3(5):683–697, 1992.

[160] L. Panait and S. Luke. Cooperative multi-agent learning: The state of the art. *Journal of Autonomous Agents and Multi-Agent Sys. (to be published)*, 2006. to be published.

[161] PC Pandey and SV Barai. Multilayer perceptron in damage detection of bridge structures. *Computers & Structures*, 54(4):597–608, 1995.

[162] Alexandros Pantelopoulos and Nikolaos G Bourbakis. Prognosis- a wearable health-monitoring system for people at risk: Methodology

and modeling. *Information Technology in Biomedicine, IEEE Transactions on*, 14(3):613–621, 2010.

[163] Scott M Pappada, Brent D Cameron, Paul M Rosman, Raymond E Bourey, Thomas J Papadimos, William Olorunto, and Marilyn J Borst. Neural network-based real-time prediction of glucose in patients with insulin-dependent diabetes. *Diabetes technology & therapeutics*, 13(2):135–141, 2011.

[164] Joseph A Paradiso and Thad Starner. Energy scavenging for mobile and wireless electronics. *Pervasive Computing, IEEE*, 4(1):18–27, 2005.

[165] Rafael S Parpinelli and Heitor S Lopes. New inspirations in swarm intelligence: a survey. *International Journal of Bio-Inspired Computation*, 3(1):1–16, 2011.

[166] Pravin Pawar, Val Jones, Bert-Jan F Van Beijnum, and Hermie Hermens. A framework for the comparison of mobile patient monitoring systems. *Journal of biomedical informatics*, 45(3):544–556, 2012.

[167] R. U. Pedersen. Tut. o. using support vector machines and java to create ambient information system. *EUSAI*, Nov 2004(Nov 2004), November 2004.

[168] Carmen Pérez-Gandía, A Facchinetti, G Sparacino, C Cobelli, EJ Gómez, M Rigla, A De Leiva, and ME Hernando. Artificial neural network algorithm for online glucose prediction from

continuous glucose monitoring. *Diabetes technology & therapeutics*, 12(1):81–88, 2010.

[169] C. Perlich, F. Provost, and J. S. Simonoff. Tree induction vs. logistic regression: A learning curve analysis. *Journal of Machine Learning Research*, (4 (2003)):211–255, 2003.

[170] Gian Paolo Perrucci, Frank HP Fitzek, and Jörg Widmer. Survey on energy consumption entities on the smartphone platform. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pages 1–6. IEEE, 2011.

[171] Kemal Polat and Salih Güneş. A novel hybrid intelligent method based on c4. 5 decision tree classifier and one-against-all approach for multi-class classification problems. *Expert Systems with Applications*, 36(2):1587–1592, 2009.

[172] Dick Pountain. Run-length encoding. *Byte*, 12(6):317–319, 1987.

[173] J Puentes, RK Bali, N Wickramasinghe, and RNG Naguib. Telemedicine trends and challenges: a technology management perspective. *International Journal of Biomedical Engineering and Technology*, 1(1):59–72, 2007.

[174] John Ross Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[175] Rashedur M Rahman and Fazle Rabbi Md Hasan. Using and comparing different decision tree classification techniques for

mining icddr, b hospital surveillance data. *Expert Systems with Applications*, 38(9):11421–11436, 2011.

[176] S.M. Redl, M.K. Weber, and M. W. Oliphant. *An Introduction to GSM*. Artech House, 1995.

[177] Attila Reiss, Didier Stricker, and Ilias Lamprinos. An integrated mobile system for long-term aerobic activity monitoring and support in daily life. In *Trust, Security and Privacy in Computing and Communications (TrustCom), 2012 IEEE 11th International Conference on*, pages 2021–2028. IEEE, 2012.

[178] Yonglin Ren, Richard Werner Nelem Pazzi, and Azzedine Boukerche. Monitoring patients via a secure and mobile healthcare system. *Wireless Communications, IEEE*, 17(1):59–65, 2010.

[179] LEE Ren-Guey, Chun-Chieh Hsiao, CHEN Chun-Chung, and LIU Ming-Shiu. A mobile-care system integrated with bluetooth blood pressure and pulse monitor, and cellular phone. *IEICE transactions on information and systems*, 89(5):1702–1711, 2006.

[180] J. Rodrigues, O. Pereira, and P. Neves. Biofeedback data visualization for body sensor networks. *Journal of Network and Computer Applications*, 34:151–158, January 2011. http://www.sciencedirect.com/science/article/pii/S1084804510001530.

[181] J. Rodriguez, A. Goni, and A. Illarramendi. Real-time classification of ecgs on a pda. In *IEEE Transactions on Information Technology in Biomedicine*, volume 9, pages 23–34. IEEE, 2005.

[182] P. Rong and M.; Pedram. Extending lifetime of a network of battery-powered mobile devices by remote processing. *Proc. of the 40th conference on Design automation*, (June 2-6, 2003):906–911, 2003.

[183] Dennis W Ruck, Steven K Rogers, Matthew Kabrisky, Mark E Oxley, and Bruce W Suter. The multilayer perceptron as an approximation to a bayes optimal discriminant function. *Neural Networks, IEEE Transactions on*, 1(4):296–298, 1990.

[184] A. Rudenko, P. Reiher, G. Popek, and G. Kuenning. Saving portable computer battery power through remote process execution. *Mobile Computing and Communications Review*, 2(1):19–26, 1998.

[185] P. Rudenko, A. and; Reiher, G. Popek, and G. Kuenning. The remote processing framework for portable computer power saving. *Proc. of the '99 Symposium on Applied Computing*, pages 365–372, 1999.

[186] T.A. Russ. Use of data abstraction methods to simplify monitoring. *Artificial Intelligence in Medicine*, (7 (1995)):497–514, 1995. Elsevier.

[187] N.C. Sahoo, S. Ganguly, and D. Das. Multi-objective planning of electrical distribution systems incorporating sectionalizing switches and tie-lines using particle swarm optimization. *Swarm and Evolutionary Computation*, 3:15–32, 2012.

[188] Antun Samukic. Umts universal mobile telecommunications system: development of standards for the third generation.

In *Global Telecommunications Conference, 1998. GLOBECOM 1998. The Bridge to Global Integration. IEEE*, volume 4, pages 1976–1983. IEEE, 1998.

[189] Stefan Schliebs and Nikola Kasabov. Evolving spiking neural networkÑa survey. *Evolving Systems*, 4(2):87–98, 2013.

[190] T. L. Seng, M. Khalid, and R. Yusof. Tuning of a neuro-fuzzy controller by genetic algorithm. *IEEE Transactions on Systems*, April 1999, April 1999.

[191] Aman Kumar Sharma and Suruchi Sahni. A comparative study of classification algorithms for spam email data analysis. *International Journal on Computer Science and Engineering*, 3(5):1890–1895, 2011.

[192] Krishnendu Shaw, Ravi Shankar, Surendra S. Yadav, and Lakshman S. Thakur. Supplier selection using fuzzy ahp and fuzzy multi-objective linear programming for developing low carbon supply chain. *Expert Systems with Applications*, 39:8182–8192, 2012.

[193] Nathan S Shenck and Joseph A Paradiso. Energy scavenging with shoe-mounted piezoelectrics. *Ieee Micro*, 21(3):30–42, 2001.

[194] Claude Sicotte, Guy Pare, Sandra Morin, Jacques Potvin, and Marie-Pierre Moreault. Effects of home telemonitoring to support improved care for chronic obstructive pulmonary diseases. *Telemedicine and e-Health*, 17(2):95–103, 2011.

[195] Sylvia Siebig, Silvia Kuhls, Michael Imhoff, Ursula Gather, Jürgen Schölmerich, and Christian E Wrede. Intensive care unit alarmsÑhow many do we need?*. *Critical care medicine*, 38(2):451–456, 2010.

[196] A.J. Sierra. Recursive and non-recursive method to serialize object at j2me. 2006.

[197] K. Sindhya, K. Miettinen, and K. Deb. A hybrid framework for evolutionary multi-objective optimization. *Evolutionary Computation, IEEE Transactions on*, 17(4):495–511, Aug 2013.

[198] James F. Smith. Evolving fuzzy decision tree structure that adapts in real time. *GECCO '05*, (June 25-29):1737–1744, June 2005. ACM.

[199] Ping Jack Soh, Bertold Van den Bergh, Hantao Xu, Hadi Aliakbarian, Saeed Farsi, Purna Samal, Guy AE Vandenbosch, Dominique Schreurs, and BKJC Nauwelaers. A smart wearable textile array system for biomedical telemetry applications. *IEEE Transactions on Microwave Theory and Techniques*, pages 1–9, 2013.

[200] Kenneth O Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary computation*, 10(2):99–127, 2002.

[201] Kenneth O Stanley and Risto Miikkulainen. Competitive coevolution through evolutionary complexification. *J. Artif. Intell. Res.(JAIR)*, 21:63–100, 2004.

[202] Andrew S. Tanenbaum and Maarten Van Steen. *Distributed Systems - Principles and Paradigms*. Prentice hall, 2002. 0-13-121786-0.

[203] Hasan Temurtas, Nejat Yumusak, and Feyzullah Temurtas. A comparative study on diabetes disease diagnosis using neural networks. *Expert Systems with applications*, 36(4):8610–8615, 2009.

[204] C.L. Tsien. Trendfinder: Automated detection of alarmable trends. *PhD Thesis*, June 2000. M.I.T.

[205] L.H. Tsoukalas. Neurofuzzy approaches to anticipation: A new paradigm for intelligent systems. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 28(4):573–582, August 1998.

[206] Navin Kumar Tyagi, AK Solanki, and Sanjay Tyagi. An algorithmic approach to data preprocessing in web usage mining. *International Journal of Information Technology and Knowledge Management*, 2(2):279–283, 2010.

[207] Andreas Valentin and Patrick Ferdinande. Recommendations on basic requirements for intensive care units: structural and organizational aspects. *Intensive care medicine*, 37(10):1575–1587, 2011.

[208] Ekhiotz Jon Vergara and Simin Nadjm-Tehrani. Energy-aware cross-layer burst buffering for wireless communication. In

Proceedings of the 3rd International Conference on Future Energy Systems: Where Energy, Computing and Communication Meet, page 24. ACM, 2012.

[209] Hariharasudhan Viswanathan, Baozhi Chen, and Dario Pompili. Research challenges in computation, communication, and context awareness for ubiquitous healthcare. *Communications Magazine, IEEE*, 50(5):92–99, 2012.

[210] NK Vuong, S Chan, CT Lau, and KM Lau. A predictive location-aware algorithm for dementia care. In *Consumer Electronics (ISCE), 2011 IEEE 15th International Symposium on*, pages 339–342. IEEE, 2011.

[211] Jun Wang, Beijun Shen, and Yuting Chen. Compressed c4. 5 models for software defect prediction. In *Quality Software (QSIC), 2012 12th International Conference on*, pages 13–16. IEEE, 2012.

[212] Xin Wang. Online time-series prediction system: Efunn-t. *Department of Information Science, University of Otago, Dunedin, New Zealand*, 2001(?). xinw@infoscience.otago.ac.nz.

[213] M. Watts and N. Kasabov. Simple evolving connectionist systems and experiments on isolated phoneme recognition. *Combinations of Evolutionary Computation and Neural Networks, 2000 IEEE Symposium*, pages 232–239, 2000.

[214] M. J. Watts and S.P. Worner. Comparison of a self organising map and simple evolving connectionist system for predicting insect pest

establishment. *International Journal of Information Technology*, 12(6):35–42, 2006.

[215] Michael Watts and Nik Kasabov. Evolutionary optimisation of evolving connectionist systems. In *Computational Intelligence, Proceedings of the World on Congress on*, volume 1, pages 606–610. IEEE, 2002.

[216] M.J. Watts. Nominal-scale evolving connectionist systems. *Neural Networks, 2006. IJCNN '06. International Joint Conference on Neural Networks*, pages 4057–4061, 2006.

[217] A. S. Weigend and N. A. (Eds.) Gershenfeld. *Time Series Prediction: Forecasting the Future and Understanding the Past.* Addison-Wesley, 1994.

[218] Darrell West. How mobile devices are transforming healthcare. *Issues in technology innovation*, 18:1–14, 2012.

[219] Darrell M West. Improving health care through mobile medical devices and sensors. 2013.

[220] Mark F Wood and Scott A DeLoach. An overview of the multiagent systems engineering methodology. In *Agent-Oriented Software Engineering*, pages 207–221. Springer, 2001.

[221] B. J. Woodford. Rule extraction from spatial data using local learning techniques. *SIRC 2005- the 17th annual collegium of the spatial information research centre*, 2005.

[222] B.J. Woodford, D. Deng, and G.L. Benwell. A wavelet-based neuro-fuzzy system for data mining small image sets. *ACSW Frontiers*, 32:139–143, 2004.

[223] Michał Woźniak, Manuel Graña, and Emilio Corchado. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, 16:3–17, 2014.

[224] Simei Gomes Wysoski, Lubica Benuskova, and Nikola Kasabov. On-line learning with structural adaptation in a network of spiking neurons for visual pattern recognition. In *Artificial Neural Networks–ICANN 2006*, pages 61–70. Springer, 2006.

[225] Ming Yang and Jing Sun. Applied research and development of new smart clothing material. In Runliang Dou, editor, *Proceedings of 2012 3rd International Asia Conference on Industrial Engineering and Management Innovation (IEMI2012)*, pages 739–746. Springer Berlin Heidelberg, 2013.

[226] G.N. Yannakakis. *AI in Computer Games: Generating Interesting Interactive Opponents by the use of Evolutionary Computation*. PhD thesis, The University of Edinburgh, 2005.

[227] Xin Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

[228] Xin Yao. The evolution of evolutionary computation. In *ICAIS*, page 4, 2011.

[229] Xin Yao. Unpacking and understanding evolutionary algorithms. In *Advances in Computational Intelligence*, pages 60–76. Springer Berlin Heidelberg, 2012.

[230] Xin Yao and Yong Xu. Recent advances in evolutionary computation. *Journal of Computer Science and Technology*, 21(1):1–18, 2006.

[231] York Health Economics Consortium. Telecare for people with dementia: Evaluation of renfrewshire project. final evaluation report. 2013. University of York,Scottish Centre for Telehealth & Telecare, Joint Improvement Team.

[232] Lance Brendan Young, Paul S Chan, Xin Lu, Brahmajee K Nallamothu, Comilla Sasson, and Peter M Cram. Impact of telemedicine intensive care unit coverage on patient outcomes: a systematic review and meta-analysis. *Archives of internal medicine*, 171(6):498–506, 2011.

[233] Yang Yu, Xin Yao, and Zhi-Hua Zhou. Evolutionary algorithms as guaranteed approximation optimizers. Technical report, 2010.

[234] C. Zanchettin, F.L. Minku, and T. B. Ludermir. Design of experiments in neuro-fuzzy systems. *Hybrid Intelligent Systems, 2005. Fifth International Conference on Hybrid Intelligent Systems*, 2005.

[235] Chiara Zecchin, Andrea Facchinetti, Giovanni Sparacino, Giuseppe De Nicolao, and Claudio Cobelli. Neural network incorporating

meal information improves accuracy of short-time prediction of glucose concentration. *Biomedical Engineering, IEEE Transactions on*, 59(6):1550–1560, 2012.

[236] Jufen Zhang, Kevin M Goode, Alan Rigby, Aggie HMM Balk, and John G Cleland. Identifying patients at risk of death or hospitalisation due to worsening heart failure using decision tree analysis: Evidence from the trans-european network-home-care management system (ten-hms) study. *International journal of cardiology*, 163(2):149–156, 2013.

[237] Ying Zhang. Real time analysis of physiological data and development of alarm algorithms in the intensive care unit. *Master Thesis*, August 2003. M.I.T.; Dept. of Electrical Engineering and Computer Science.

[238] Ying Zhang, Hongbo Wang, and Shiduan Cheng. A method for real-time peer-to-peer traffic classification based on c4. 5. In *Communication Technology (ICCT), 2010 12th IEEE International Conference on*, pages 1192–1195. IEEE, 2010.

[239] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary computation*, 8(2):173–195, 2000.

[240] David Zweig and Jane Webster. Where is the line between benign and invasive? an examination of psychological barriers to the acceptance of awareness monitoring systems. *Journal of Organizational Behavior*, 23(5):605–633, 2002.

Appendix A

Measured Results Table

During the experiments, a wide range of measurements were collected for evaluation and further usage. Finally, only a selected subset of representative measurements was used to compare the different approaches. For completeness, the full results table containing all measured values throughout all experiments can be found in the following:

ResultTable

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	oaCyc datasetAvailable	oaCyc xferOverSocket	oaCyc xferComplete	oaCyc processDataset
11.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	10981464297092	680512903298	574821370324	8225725721290
11.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	21007226177675	260745860016	1315747651870	18522309073135
12.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	10986394427946	684050530899	573668421655	8228242786806
13.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	21024213289264	262206109667	1312987143824	18546189732256
16.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	yes	3630244750948	686331316850	573256232776	2531502290514
16.03.09	pat_B_default_large.xls	10	1000	1001	1	CS	yes	7696208284598	211883955947	661993174770	6443693595961
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	yes	236256930360	147434606797	128623102727	149590330911
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	CS	yes	612747960293	143505859506	133703236241	493024555649
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	yes	643270762163	310786702713	262441279540	375311167754
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	CS	yes	1871727122662	153853201277	140343021511	1488146127538
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	yes	1895722404901	315711899482	262962171366	1367615578513
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	CS	yes	3110429358026	155822920066	142690154843	2465073766882
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	yes	2357782255228	505721667351	417709746910	1630370611656
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	CS	yes	4686060226599	183847923755	316236366051	3810178098446
19.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	no	339258584506	176547425069	128409727058	148513281582
19.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	no	494147163102	216767757923	134534251601	147702794429
19.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	no	645201307327	258649042179	143513902013	150219113224
19.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	no	800166270733	304523021501	147483496058	150582764928
19.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	no	1019304221140	367675906394	160140814263	149496017578
23.03.09	pat_B_default_large.xls	3	1000	1001	1	MA	no	292965933113	276480855781	64349564641	10380036023
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	MA	no	219532774343	198482653668	54104340291	10530866284
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	MA	no	166138152359	152792423599	48163198885	10575776439
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	MA	no	112329739718	102135044464	41742036209	10427480006
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	MA	no	58770093308	53835972757	34620639393	10538457718
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	CS	no	214790050849	43942702152	79549886721	164543091192
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	CS	no	694220294032	64799621442	110952333612	547161492156
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	CS	no	1213285096843	84034811058	158329212410	983878035893
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	CS	no	1844385601517	120108538817	295550459675	1526098063212
24.03.09	pat_B_default_large.xls	3	1000	1001	1	CS	no	3046262354370	167149540787	506651603581	2594637155574

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	oaCyc datasetAvailable	oaCyc xferOverSocket	oaCyc xferComplete	oaCyc processDataset
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	yes	1865400432605	175093933352	115154645679	986083976596
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	701	*0,7*	MA	yes	1019220954428	175950139075	115415947327	503151452244
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	501	*0,5*	MA	yes	885414043534	106379663108	74461063353	447858134173
25.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	yes	297936159032	108944962234	74067104546	127988277226
25.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	yes	38332606878	47685114174	35319149226	10582260333
26.03.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	yes	7339912063620	868202375603	804756995307	6483165138834
27.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	yes	3989358835472	867022783459	800286229751	3537667093263
27.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	yes	3467240772488	556242198369	524730131422	3163203115399
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	yes	1454630259992	558421104328	525446031043	1268676607604
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	yes	72431696447	278741218101	260714363509	9317638759
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	no	289831352261	334581848732	261369567423	9312616850
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	no	413460502487	375350715543	268809153909	9311295597
31.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	no	540188970275	418528345882	278422774612	9268797089
31.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	no	660878165479	459691201930	289890406300	9327341522
01.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	no	835952699165	526020385496	304474663132	9296746125
02.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	CS	no	19776221608356	467332954905	4005923195334	19000346591908
02.04.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	CS	no	11919128479806	405690080090	2717276750248	11364263624565
02.04.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	CS	no	7598252726854	356761696392	1849242998143	7202565567921
06.04.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	CS	no	3702174385041	309488850843	1132467705303	3484566532567
06.04.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	CS	no	1130468581471	279287044011	588478231136	1047951684549
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	MA	yes	3732313766287	149159094487	149222149432	2499773420524
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	CS	no	9306263509421	198257049810	1069911770871	8440361021176
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	MA	yes	4001212401795	132551033682	145068013976	2675441249773
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	CS	yes	5350706603112	87860836790	367513877370	4622167910156
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	MA	yes	3481303028003	46774236180	64947981542	2774534389629
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	CS	yes	3492818577390	45815496481	65053224837	2788649369069
26.05.09	test_active_1.xls	10	1000	1001		MA	yes	5833532066733	154999299334	131526446780	4391291378443
27.05.09	test_active_1.xls	10	1000	1001		CS	-	6491952009186	151737891246	433388157537	5030193619112
27.05.09	test_active_2.xls	10	1000	1001		MA	yes	5836460299341	153751411497	130443330492	4376680270887

date	database_name	Dataset width	Data base size	datasets process ed	Numb er of loops	mode	State caching	oaCyc datasetAvailable	oaCyc xferOverSocket	oaCyc xferComplete	oaCyc processDataset
27.05.09	test_active_2.xls	10	1000	1001		CS	-	6543529933572	157870845197	435395670012	5056319683503
03.06.09	test_active_3.xls	10	1000	1001		MA	yes	5940006171713	322426672689	282684591940	4449430770074
03.06.09	test_active_3.xls	10	1000	1001		CS	-	8116295335728	151782494902	436180036145	6682968118078
05.06.09	test_active_4.xls	10	1000	1001		MA	yes	5752638217745	320152474096	283276504908	4277018305467
08.06.09	test_active_4.xls	10	1000	1001		CS	-	7486249944273	153335709195	435075425757	6013911567500
08.06.09	test_active_5.xls	10	1000	1001		MA	yes	5516540854949	322164562621	283504548361	4103584604303
08.06.09	test_active_5.xls	10	1000	1001		CS	-	6744845060072	150889831029	436428614170	5267770300506
09.06.09	test_active_6.xls	10	1000	1001		MA	yes	6433534461093	320489727321	284070775060	4809706193298
09.06.09	test_active_6.xls	10	1000	1001		CS	-	9437835138975	151585022122	436296887137	8031990390951
12.06.09	test_active_7.xls	10	1000	1001		MA	yes	5818159529668	153851250650	130585105175	4393722902639
12.06.09	test_active_7.xls	10	1000	1001		CS	-	6505279377855	151437910349	436869135116	5054853265930
15.06.09	test_active_8.xls	10	1000	1001		MA	yes	5616535030103	334025805627	282647904037	4205088414819
15.06.09	test_active_8.xls	10	1000	1001		CS	-	6881436327453	150975213597	438792681081	5730201022159
16.06.09	test_active_9.xls	10	1000	1001		MA	yes	5623426978158	161725696003	131819221203	4191731850182
16.06.09	test_active_9.xls	10	1000	1001		CS	-	6677827704944	151865669044	436133103521	5197785081456
16.06.09	test_active_10.xls	10	1000	1001		MA	yes	5660327532860	321677056282	282445902929	4227916581966
16.06.09	test_active_10.xls	10	1000	1001		CS	-	7289005381355	151714678770	435442532128	5832873505203
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes	5236988662206	502166568510	440343452106	3857795392633
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-	8183196354785	172220872888	600392285630	6759666161544
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes	5180500361253	497414458147	432623282823	3812225191629
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-	8048079539004	170211514176	639729944763	6894406909938
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes	5209646960286	504616398939	434207330870	3845937634577
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-	8291552575128	178053949353	623604509488	6853589275107
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes	6003071061143	316812381467	279890946188	4540054440622
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-	8277786619339	169022051058	619673344033	6837224374406
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes	4836042852749	514407496750	437961432841	3505865320583
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-	8256370640219	176382206491	610175986275	6840158884585
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes	5320857484539	513489070426	437495540030	3939652440119
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-	8353537768629	173498661855	638580489796	6906840451442

date	database_name	Dataset width	Data base size	datasets process ed	Numb er of loops	mode	State caching	oaCyc datasetAvailable	oaCyc xferOverSocket	oaCyc xferComplete	oaCyc processDataset
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes	6031490769495	322261192842	282626077856	4593262852241
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-	8321233637376	174709468218	633133136174	6891993481128
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes	5298175197219	507981730045	437438090472	3906131749782
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-	8292549294957	176402748861	608513033822	6860112757491
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes	5252996979854	504379642143	435302039864	3878266855486
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-	8369619985623	170212061560	639192192845	6938401892569
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes	5231265663837	500618617337	434978384663	3863810087082
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-	8289918388053	165064701479	634142320435	6869147414774

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	perDsCyc datasetAvailable	PerDsCyc xferOverSocket	PerDsCyc xferComplete	perDsCyc processDataset
11.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	5487988154,47	340086408,44	287267051,64	4110807456,92
11.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	10498363906,88	130307776,12	657545053,41	9256526273,43
12.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	5490451987,98	341854338,28	286690865,39	4112065360,72
13.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	10506853218,02	131037536,07	656165489,17	9268460635,81
16.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	yes	3626618132,82	685645671,18	572683549,23	2528973317,2
16.03.09	pat_B_default_large.xls	10	1000	1001	1	CS	yes	7688519764,83	211672283,66	661331842,93	6437256339,62
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	yes	2339177528,32	1459748582,15	1273496066,6	1481092385,26
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	CS	yes	6066811488,05	1420850094,12	1323794418,23	4881431244,05
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	yes	2137112166,65	1032513962,5	871897938,67	1246880955,99
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	CS	yes	6218362533,76	511140203,58	466255885,42	4944007068,23
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	yes	3783877055,69	630163472,02	524874593,54	2729771613,8
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	CS	yes	6208441832,39	311023792,55	284810688,31	4920306919,92
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	yes	3363455428,29	721428912,06	595876957,08	2325778333,32
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	CS	yes	6684822006,56	262265226,47	451121777,53	5435346788,08
19.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	no	3358995886,2	1747994307,61	1271383436,22	1470428530,51
19.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	no	1641684927,25	720158664,2	446957646,51	490706958,24
19.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	no	1287826960,73	516265553,25	286454894,24	299838549,35
19.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	no	1141464009,6	434412298,86	210390151,3	214811362,24
19.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	no	1018285935,2	367308597,8	159980833,43	149346670,91
23.03.09	pat_B_default_large.xls	3	1000	1001	1	MA	no	292673259,85	276204651,13	64285279,36	10369666,36
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	MA	no	313170862,12	283142159,3	77181655,19	15022633,79
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	MA	no	331613078,56	304974897,4	96134129,51	21109334,21
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	MA	no	373188504,05	339319084,6	138677861,16	34642790,72
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	MA	no	581882111,96	533029433,24	342778607,85	104341165,52
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	CS	no	2126634166,82	435076258,93	787622640,8	1629139516,75
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	CS	no	2306379714,39	215281134,36	368612404,03	1817812266,3
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	CS	no	2421726740,21	167734153,81	316026372,08	1963828414,96
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	CS	no	2631077890,89	171338857,09	421612638,62	2177030047,38
24.03.09	pat_B_default_large.xls	3	1000	1001	1	CS	no	3043219135,23	166982558,23	506145458,12	2592045110,46

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	perDsCyc datasetAvailable	PerDsCyc xferOverSocket	PerDsCyc xferComplete	perDsCyc processDataset
24.03.09	pat_B_default_large.xls	3	1000	1001	1	MA	yes	1863536895,71	174919014,34	115039606,07	985098877,72
24.03.09	pat_B_default_large.xls	3	1000	701	0,7	MA	yes	1453952859,38	250998771,86	164644718,01	717762414,04
24.03.09	pat_B_default_large.xls	3	1000	501	0,5	MA	yes	1767293500,07	212334656,9	148624876,95	893928411,52
25.03.09	pat_B_default_large.xls	3	1000	301	0,3	MA	yes	989821126,35	361943396,13	246070114,77	425210223,34
25.03.09	pat_B_default_large.xls	3	1000	101	0,1	MA	yes	379530761,17	472129843,31	349694546,79	104774854,78
26.03.09	pat_B_default_large.xls	7	1000	1001	1	MA	yes	7332579484,14	867335040,56	803953042,26	6476688450,38
27.03.09	pat_B_default_large.xls	7	1000	701	0,7	MA	yes	5690954116,22	1236837066,28	1141635135,17	5046600703,66
27.03.09	pat_B_default_large.xls	7	1000	501	0,5	MA	yes	6920640264,45	1110263869	1047365531,78	6313778673,45
31.03.09	pat_B_default_large.xls	7	1000	301	0,3	MA	yes	4832658671,07	1855219615,71	1745667877,22	4214872450,51
31.03.09	pat_B_default_large.xls	7	1000	101	0,1	MA	yes	717145509,38	2759814040,6	2581330331,77	92253849,1
31.03.09	pat_B_default_large.xls	7	1000	101	0,1	MA	no	2869617349,12	3312691571,6	2587817499,24	92204127,23
31.03.09	pat_B_default_large.xls	7	1000	301	0,3	MA	no	1373622931,85	1247012344	893053667,47	30934536,87
31.03.09	pat_B_default_large.xls	7	1000	501	0,5	MA	no	1078221497,55	835385919,92	555734081,06	18500592,99
31.03.09	pat_B_default_large.xls	7	1000	701	0,7	MA	no	942764858,03	655764910,03	413538382,74	13305765,37
01.04.09	pat_B_default_large.xls	7	1000	1001	1	MA	no	835117581,58	525494890,61	304170492,64	9287458,67
02.04.09	pat_B_default_large.xls	7	1000	1001	1	CS	no	19756465143,21	466866088,82	4001921274,06	18981365226,68
02.04.09	pat_B_default_large.xls	7	1000	701	0,7	CS	no	17003036347,8	578730499,42	3876286376,96	16211503030,76
02.04.09	pat_B_default_large.xls	7	1000	501	0,5	CS	no	15166173107,49	712099194,4	3691103788,71	14376378379,08
06.04.09	pat_B_default_large.xls	7	1000	301	0,3	CS	no	12299582674,55	1028202162,27	3762351180,41	11576632998,56
06.04.09	pat_B_default_large.xls	7	1000	101	0,1	CS	no	11192758232,39	2765218257,53	5826517139,96	10375759252,96
06.04.09	pat_B_default_small.xls	10	300	1001	3,34	MA	yes	3728585181,11	149010084,4	149073076,36	2497276144,38
06.04.09	pat_B_default_small.xls	10	300	1001	3,34	CS	no	9296966542,88	198058990,82	1068842927,94	8431929092,08
08.04.09	pat_C_default_small.xls	10	300	1001	3,34	MA	yes	3997215186,61	132418615,07	144923090,89	2672768481,29
08.04.09	pat_C_default_small.xls	10	300	1001	3,34	CS	yes	5345361241,87	87773063,73	367146730,64	4617550359,8
09.04.09	pat_D_default_small.xls	10	300	1001	3,34	MA	yes	3477825202,8	46727508,67	64883098,44	2771762627
09.04.09	pat_D_default_small.xls	10	300	1001	3,34	CS	yes	3489329248,14	45769726,75	64988236,6	2785863505,56
26.05.09	test_active_1.xls	10	1000	1001		MA	yes				
27.05.09	test_active_1.xls	10	1000	1001		CS	-				
27.05.09	test_active_2.xls	10	1000	1001		MA	yes				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	perDsCyc datasetAvailable	PerDsCyc xferOverSocket	PerDsCyc xferComplete	perDsCyc processDataset
27.05.09	test_active_2.xls	10	1000	1001		CS	-				
03.06.09	test_active_3.xls	10	1000	1001		MA	yes				
03.06.09	test_active_3.xls	10	1000	1001		CS	-				
05.06.09	test_active_4.xls	10	1000	1001		MA	yes				
08.06.09	test_active_4.xls	10	1000	1001		CS	-				
08.06.09	test_active_5.xls	10	1000	1001		MA	yes				
08.06.09	test_active_5.xls	10	1000	1001		CS	-				
09.06.09	test_active_6.xls	10	1000	1001		MA	yes				
09.06.09	test_active_6.xls	10	1000	1001		CS	-				
12.06.09	test_active_7.xls	10	1000	1001		MA	yes				
12.06.09	test_active_7.xls	10	1000	1001		CS	-				
15.06.09	test_active_8.xls	10	1000	1001		MA	yes				
15.06.09	test_active_8.xls	10	1000	1001		CS	-				
16.06.09	test_active_9.xls	10	1000	1001		MA	yes				
16.06.09	test_active_9.xls	10	1000	1001		CS	-				
16.06.09	test_active_10.xls	10	1000	1001		MA	yes				
16.06.09	test_active_10.xls	10	1000	1001		CS	-				
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes				
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-				
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes				
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-				
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes				
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-				
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes				
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-				
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes				
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-				
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes				
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-				

date	database_name	Dataset width	Data base size	datasets process ed	Numb er of loops	mode	State caching	perDsCyc datasetAvailable	PerDsCyc xferOverSocket	PerDsCyc xferComplete	perDsCyc processDataset
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes				
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-				
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes				
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-				
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes				
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-				
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes				
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	avgClassificationTime client	avgDurLearnStep client	conns opened	conn_open time	conn_active time
11.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	2	MA	*yes*	1675	0	4	1178	216
11.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	2	CS	*yes*	2793	1014	16	3228	85
12.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	2	MA	*yes*	1676	0	4	1179	217
13.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	2	CS	*yes*	2792	1013	16	3225	86
16.03.09	*pat_B_default_large.xls*	*10*	*1000*	*1001*	*1*	MA	*yes*	1590	0	4	1179	215
16.03.09	*pat_B_default_large.xls*	*10*	*1000*	*1001*	*1*	CS	*yes*	1940	898	8	1515	67
17.03.09	*pat_B_default_large.xls*	*10*	*1000*	*101*	*0,1*	MA	*yes*	1455	0	2	191	46
17.03.09	*pat_B_default_large.xls*	*10*	*1000*	*101*	*0,1*	CS	*yes*	1469	703	2	258	43
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	*301*	*0,3*	MA	*yes*	1489	0	2	522	98
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	301	*0,3*	CS	*yes*	1487	713	3	440	47
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	MA	*yes*	1497	0	3	522	98
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	CS	*yes*	1496	715	4	438	48
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	MA	*yes*	1529	0	3	863	159
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	CS	*yes*	1642	803	6	991	58
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	101	*0,1*	MA	no	1454	0	2	230	54
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	301	*0,3*	MA	no	1458	0	2	556	68
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	MA	no	1461	0	2	882	85
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	MA	no	1463	0	2	1205	101
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	1001	*1*	MA	no	1457	0	2	1691	129
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	no	444	0	2	1108	93
23.03.09	*pat_B_default_large.xls*	3	*1000*	701	*0,7*	MA	no	451	0	2	781	65
23.03.09	*pat_B_default_large.xls*	3	*1000*	501	*0,5*	MA	no	453	0	2	563	50
23.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	no	446	0	2	346	33
24.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	no	453	0	2	128	18
24.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	CS	no	493	549	2	184	14
23.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	CS	no	546	568	3	379	22
23.03.09	*pat_B_default_large.xls*	3	*1000*	501	*0,5*	CS	no	592	566	4	561	28
23.03.09	*pat_B_default_large.xls*	3	*1000*	701	*0,7*	CS	no	657	611	7	937	41
24.03.09	*pat_B_default_large.xls*	3	*1000*	1001	*1*	CS	no	780	666	9	1469	57

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	avgClassificationTime client	avgDurLearnStep client	conns opened	conn_open time	conn_active time
24.03.09	*pat_B_default_large.xls*	3	1000	1001	1	MA	yes	484	0	3	477	56
24.03.09	*pat_B_default_large.xls*	3	1000	701	0,7	MA	yes	483	0	3	478	56
24.03.09	*pat_B_default_large.xls*	3	1000	501	0,5	MA	yes	482	0	2	277	34
25.03.09	*pat_B_default_large.xls*	3	1000	301	0,3	MA	yes	481	0	2	278	35
25.03.09	*pat_B_default_large.xls*	3	1000	101	0,1	MA	yes	453	0	2	120	15
26.03.09	*pat_B_default_large.xls*	7	1000	1001	1	MA	yes	2899	0	3	940	266
27.03.09	*pat_B_default_large.xls*	7	1000	701	0,7	MA	yes	2866	0	3	941	264
27.03.09	*pat_B_default_large.xls*	7	1000	501	0,5	MA	yes	2856	0	2	549	169
31.03.09	*pat_B_default_large.xls*	7	1000	301	0,3	MA	yes	2854	0	2	549	170
31.03.09	*pat_B_default_large.xls*	7	1000	101	0,1	MA	yes	2797	0	2	297	86
31.03.09	*pat_B_default_large.xls*	7	1000	101	0,1	MA	no	2797	0	2	376	101
31.03.09	*pat_B_default_large.xls*	7	1000	301	0,3	MA	no	2797	0	2	667	120
31.03.09	*pat_B_default_large.xls*	7	1000	501	0,5	MA	no	2797	0	2	951	135
31.03.09	*pat_B_default_large.xls*	7	1000	701	0,7	MA	no	2797	0	2	1235	153
01.04.09	*pat_B_default_large.xls*	7	1000	1001	1	MA	no	2797	0	2	1666	178
02.04.09	*pat_B_default_large.xls*	7	1000	1001	1	CS	no	5718	1529	33	8068	150
02.04.09	*pat_B_default_large.xls*	7	1000	701	0,7	CS	no	4891	1379	24	5122	129
02.04.09	*pat_B_default_large.xls*	7	1000	501	0,5	CS	no	4337	1263	17	3372	114
06.04.09	*pat_B_default_large.xls*	7	1000	301	0,3	CS	no	3796	1160	9	1622	105
06.04.09	*pat_B_default_large.xls*	7	1000	101	0,1	CS	no	3143	1078	4	701	84
06.04.09	*pat_B_default_small.xls*	10	300	1001	3,34	MA	yes	961	0	2	423	49
06.04.09	*pat_B_default_small.xls*	10	300	1001	3,34	CS	no	2547	858	20	3376	67
08.04.09	*pat_C_default_small.xls*	10	300	1001	3,34	MA	yes	924	0	2	283	42
08.04.09	*pat_C_default_small.xls*	10	300	1001	3,34	CS	yes	1391	711	5	761	28
09.04.09	*pat_D_default_small.xls*	10	300	1001	3,34	MA	yes	836	0	1	33	14
09.04.09	*pat_D_default_small.xls*	10	300	1001	3,34	CS	yes	837	0	1	34	14
26.05.09	test_active_1.xls	10	1000	1001		MA	yes	1466	0	2	236	49
27.05.09	test_active_1.xls	10	1000	1001		CS	-	1524	860	3	533	55
27.05.09	test_active_2.xls	10	1000	1001		MA	yes	1467	0	2	235	49

date	database_name	Dataset width	Data base size	datasets process ed	Numb er of loops	mode	State caching	avgClassificat ionTime client	avgDurLe arnStep client	conns opened	conn_open time	conn_active time
27.05.09	test_active_2.xls	10	1000	1001		CS	-	1522	861	3	532	52
03.06.09	test_active_3.xls	10	1000	1001		MA	yes	1596	0	2	427	101
03.06.09	test_active_3.xls	10	1000	1001		CS	-	2010	862	3	524	46
05.06.09	test_active_4.xls	10	1000	1001		MA	yes	1534	0	2	428	101
08.06.09	test_active_4.xls	10	1000	1001		CS	-	1810	862	3	525	47
08.06.09	test_active_5.xls	10	1000	1001		MA	yes	1471	0	2	427	102
08.06.09	test_active_5.xls	10	1000	1001		CS	-	1606	865	3	525	47
09.06.09	test_active_6.xls	10	1000	1001		MA	yes	1724	0	2	427	100
09.06.09	test_active_6.xls	10	1000	1001		CS	-	2441	863	3	525	46
12.06.09	test_active_7.xls	10	1000	1001		MA	yes	1467	0	2	236	49
12.06.09	test_active_7.xls	10	1000	1001		CS	-	1520	861	3	525	46
15.06.09	test_active_8.xls	10	1000	1001		MA	yes	1507	0	2	432	106
15.06.09	test_active_8.xls	10	1000	1001		CS	-	1752	870	3	539	55
16.06.09	test_active_9.xls	10	1000	1001		MA	yes	1465	0	2	301	52
16.06.09	test_active_9.xls	10	1000	1001		CS	-	1565	859	3	525	47
16.06.09	test_active_10.xls	10	1000	1001		MA	yes	1517	0	2	427	100
16.06.09	test_active_10.xls	10	1000	1001		CS	-	1754	860	3	525	47
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes	1626	0	3	723	157
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-	2037	893	5	917	53
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes	1603	0	3	719	156
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-	2115	902	6	1111	64
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes	1619	0	3	722	157
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-	2067	894	6	1086	55
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes	1621	0	2	424	99
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-	2063	899	5	951	61
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes	1604	0	4	815	160
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-	2057	891	6	1106	55
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes	1622	0	3	700	162
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-	2083	906	5	953	53

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	avgClassificationTime client	avgDurLearnStep client	conns opened	conn_open time	conn_active time
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes	1644	0	2	428	102
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-	2077	902	5	988	57
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes	1619	0	3	705	157
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-	2066	895	6	1139	55
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes	1622	0	3	707	155
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-	2087	902	5	985	53
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes	1620	0	3	717	157
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-	2076	899	4	820	51

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	conn_idle time	BytesRead client	bytesWritten client	bytesXferred	number_of_evolvingNodes clientNet
11.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	962	131005	60005	191010	134
11.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	3143	31005	89005	120010	412
12.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	962	131005	60005	191010	134
13.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	3139	31005	89005	120010	412
16.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	yes	964	131005	60005	191010	134
16.03.09	pat_B_default_large.xls	10	1000	1001	1	CS	yes	1448	30005	50005	80010	262
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	yes	145	29005	8005	37010	112
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	CS	yes	215	29005	8005	37010	114
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	yes	424	60005	26005	86010	115
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	CS	yes	393	29005	17005	46010	115
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	yes	424	60005	26005	86010	115
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	CS	yes	390	29005	17005	46010	116
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	yes	704	95005	44005	139010	134
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	CS	yes	933	29005	37005	66010	168
19.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	no	176	29005	37005	66010	112
19.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	no	488	30005	61005	91010	112
19.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	no	797	31005	85005	116010	112
19.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	no	1104	31005	108005	139010	112
19.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	no	1562	33005	143005	176010	112
23.03.09	pat_B_default_large.xls	3	1000	1001	1	MA	no	1015	9005	36005	45010	57
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	MA	no	716	8005	26005	34010	57
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	MA	no	513	7005	20005	27010	57
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	MA	no	313	6005	14005	20010	57
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	MA	no	110	5005	8005	13010	57
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	CS	no	170	5005	6005	11010	78
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	CS	no	357	5005	14005	19010	90
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	CS	no	533	5005	19005	24010	109
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	CS	no	896	6005	29005	35010	162
24.03.09	pat_B_default_large.xls	3	1000	1001	1	CS	no	1412	6005	41005	47010	235

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	conn_idle time	BytesRead client	bytesWritten client	bytesXferred	number_of_evolvingNodes clientNet
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	yes	*421*	19005	12005	*31010*	66
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	701	*0,7*	MA	yes	*422*	19005	12005	*31010*	66
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	501	*0,5*	MA	yes	*243*	12005	6005	*18010*	66
25.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	yes	*243*	12005	6005	*18010*	66
25.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	yes	*105*	5005	2005	*7010*	57
26.03.09	*pat_B_default_large.xls*	*7*	*1000*	1001	*1*	MA	yes	*674*	182005	29005	*211010*	328
27.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	yes	*677*	182005	29005	*211010*	328
27.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	yes	*380*	119005	15005	*134010*	318
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	yes	*379*	119005	15005	*134010*	318
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	yes	*211*	58005	9005	*67010*	310
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	no	*275*	58005	67005	*125010*	310
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	no	*547*	59005	85005	*144010*	310
31.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	no	*816*	60005	103005	*163010*	310
31.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	no	*1082*	61005	121005	*182010*	310
01.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	no	*1488*	62005	147005	*209010*	310
02.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	CS	no	*7918*	61005	110005	*171010*	981
02.04.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	CS	no	*4993*	60005	78005	*138010*	793
02.04.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	CS	no	*3258*	59005	56005	*115010*	649
06.04.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	CS	no	*1517*	59005	31005	*90010*	509
06.04.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	CS	no	*617*	58005	12005	*70010*	392
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	MA	yes	*374*	23005	25005	*48010*	68
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	CS	no	*3309*	12005	104005	*116010*	346
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	MA	yes	*241*	23005	15005	*38010*	67
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	CS	yes	*733*	10005	28005	*38010*	159
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	MA	yes	*19*	10005	5	*10010*	57
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	CS	yes	*20*	10005	5	*10010*	57
26.05.09	test_active_1.xls	10	1000	1001		MA	yes	186	29005	11005		112
27.05.09	test_active_1.xls	10	1000	1001		CS	-	478	29005	13005		203
27.05.09	test_active_2.xls	10	1000	1001		MA	yes	185	29005	10005		112

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	conn_idle time	BytesRead client	bytesWritten client	bytesXferred	number_of_evolvingNodes clientNet
27.05.09	test_active_2.xls	10	1000	1001		CS	-	480	29005	13005		203
03.06.09	test_active_3.xls	10	1000	1001		MA	yes	326	65004	18005		135
03.06.09	test_active_3.xls	10	1000	1001		CS	-	477	29005	13005		203
05.06.09	test_active_4.xls	10	1000	1001		MA	yes	326	65004	18005		135
08.06.09	test_active_4.xls	10	1000	1001		CS	-	477	29005	13005		203
08.06.09	test_active_5.xls	10	1000	1001		MA	yes	325	65004	18005		
08.06.09	test_active_5.xls	10	1000	1001		CS	-	478	29005	13005		203
09.06.09	test_active_6.xls	10	1000	1001		MA	yes	326	65004	18005		135
09.06.09	test_active_6.xls	10	1000	1001		CS	-	478	29005	13005		203
12.06.09	test_active_7.xls	10	1000	1001		MA	yes	186	29005	11005		112
12.06.09	test_active_7.xls	10	1000	1001		CS	-	478	29005	13005		203
15.06.09	test_active_8.xls	10	1000	1001		MA	yes	3206	65004	18005		135
15.06.09	test_active_8.xls	10	1000	1001		CS	-	484	29005	13005		203
16.06.09	test_active_9.xls	10	1000	1001		MA	yes	248	29005	15005		112
16.06.09	test_active_9.xls	10	1000	1001		CS	-	478	29005	13005		203
16.06.09	test_active_10.xls	10	1000	1001		MA	yes	327	65004	18005		135
16.06.09	test_active_10.xls	10	1000	1001		CS	-	478	29005	13005		203
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes	566	100005	33005		136
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-	863	29005	24005		247
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes	563	99005	33005		133
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-	1046	29005	29005		255
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes	565	100005	33005		135
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-	1030	30005	30005		253
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes	325	64004	18005		133
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-	889	29005	25005		252
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes	655	100005	40005		134
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-	1051	30005	30005		249
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes	538	100005	31005		135
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-	899	29005	24005		256

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	conn_idle time	BytesRead client	bytesWritten client	bytesXferred	number_of_evolvingNodes clientNet
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes	326	65004	18005		136
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-	930	29005	25005		255
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes	548	100005	31005		135
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-	1083	30005	31005		248
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes	552	100005	32005		135
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-	931	29005	25005		257
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes	559	100005	32005		135
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-	768	29005	20005		256

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	ClientTrain Epochs	BackendTrain Epochs	minDurationOf ControlMillis	backendGender Param
11.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	MA	*yes*	*90*	*100*	*180000*	*MALE*
11.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	CS	*yes*	*90*	*100*	*180000*	*MALE*
12.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	MA	*yes*	*90*	*100*	*180000*	*MALE*
13.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	CS	*yes*	*90*	*100*	*180000*	*MALE*
16.03.09	*pat_B_default_large.xls*	*10*	*1000*	*1001*	*1*	MA	*yes*	*90*	*100*	*180000*	*MALE*
16.03.09	*pat_B_default_large.xls*	*10*	*1000*	*1001*	*1*	CS	*yes*	*90*	*100*	*180000*	*MALE*
17.03.09	*pat_B_default_large.xls*	*10*	*1000*	*101*	*0,1*	MA	*yes*	*90*	*100*	*180000*	*MALE*
17.03.09	*pat_B_default_large.xls*	*10*	*1000*	*101*	*0,1*	CS	*yes*	*90*	*100*	*180000*	*MALE*
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	*301*	*0,3*	MA	*yes*	*90*	*100*	*180000*	*MALE*
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	301	*0,3*	CS	*yes*	*90*	*100*	*180000*	*MALE*
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	MA	*yes*	*90*	*100*	*180000*	*MALE*
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	CS	*yes*	*90*	*100*	*180000*	*MALE*
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	MA	*yes*	*90*	*100*	*180000*	*MALE*
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	CS	*yes*	*90*	*100*	*180000*	*MALE*
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	101	*0,1*	MA	no	*90*	*100*	*180000*	*MALE*
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	301	*0,3*	MA	no	*90*	*100*	*180000*	*MALE*
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	MA	no	*90*	*100*	*180000*	*MALE*
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	MA	no	*90*	*100*	*180000*	*MALE*
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	1001	*1*	MA	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	3	*1000*	701	*0,7*	MA	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	3	*1000*	501	*0,5*	MA	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	no	*90*	*100*	*180000*	*MALE*
24.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	no	*90*	*100*	*180000*	*MALE*
24.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	CS	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	CS	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	3	*1000*	501	*0,5*	CS	no	*90*	*100*	*180000*	*MALE*
23.03.09	*pat_B_default_large.xls*	3	*1000*	701	*0,7*	CS	no	*90*	*100*	*180000*	*MALE*
24.03.09	*pat_B_default_large.xls*	3	*1000*	1001	*1*	CS	no	*90*	*100*	*180000*	*MALE*

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	ClientTrain Epochs	BackendTrain Epochs	minDurationOf ControlMillis	backendGender Param
24.03.09	*pat_B_default_large.xls*	3	*1000*	1001	*1*	MA	yes	*90*	*100*	*180000*	*MALE*
24.03.09	*pat_B_default_large.xls*	3	*1000*	701	*0,7*	MA	yes	*90*	*100*	*180000*	*MALE*
24.03.09	*pat_B_default_large.xls*	3	*1000*	501	*0,5*	MA	yes	*90*	*100*	*180000*	*MALE*
25.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	yes	*90*	*100*	*180000*	*MALE*
25.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	yes	*90*	*100*	*180000*	*MALE*
26.03.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	yes	*90*	*100*	*180000*	*MALE*
27.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	yes	*90*	*100*	*180000*	*MALE*
27.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	yes	*90*	*100*	*180000*	*MALE*
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	yes	*90*	*100*	*180000*	*MALE*
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	yes	*90*	*100*	*180000*	*MALE*
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	no	*90*	*100*	*180000*	*MALE*
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	no	*90*	*100*	*180000*	*MALE*
31.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	no	*90*	*100*	*180000*	*MALE*
31.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	no	*90*	*100*	*180000*	*MALE*
01.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	no	*90*	*100*	*180000*	*MALE*
02.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	CS	no	*90*	*100*	*180000*	*MALE*
02.04.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	CS	no	*90*	*100*	*180000*	*MALE*
02.04.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	CS	no	*90*	*100*	*180000*	*MALE*
06.04.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	CS	no	*90*	*100*	*180000*	*MALE*
06.04.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	CS	no	*90*	*100*	*180000*	*MALE*
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	MA	yes	*90*	*100*	*180000*	*MALE*
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	CS	no	*90*	*100*	*180000*	*MALE*
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	MA	yes	*90*	*100*	*180000*	*MALE*
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	CS	yes	*90*	*100*	*180000*	*MALE*
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	MA	yes	*90*	*100*	*180000*	*MALE*
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	CS	yes	*90*	*100*	*180000*	*MALE*
26.05.09	test_active_1.xls	10	1000	1001		MA	yes	*90*	*100*	*180000*	*MALE*
27.05.09	test_active_1.xls	10	1000	1001		CS	-				
27.05.09	test_active_2.xls	10	1000	1001		MA	yes				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	ClientTrain Epochs	BackendTrain Epochs	minDurationOf ControlMillis	backendGender Param
27.05.09	test_active_2.xls	10	1000	1001		CS	-				
03.06.09	test_active_3.xls	10	1000	1001		MA	yes				
03.06.09	test_active_3.xls	10	1000	1001		CS	-				
05.06.09	test_active_4.xls	10	1000	1001		MA	yes				
08.06.09	test_active_4.xls	10	1000	1001		CS	-				
08.06.09	test_active_5.xls	10	1000	1001		MA	yes				
08.06.09	test_active_5.xls	10	1000	1001		CS	-				
09.06.09	test_active_6.xls	10	1000	1001		MA	yes				
09.06.09	test_active_6.xls	10	1000	1001		CS	-				
12.06.09	test_active_7.xls	10	1000	1001		MA	yes				
12.06.09	test_active_7.xls	10	1000	1001		CS	-				
15.06.09	test_active_8.xls	10	1000	1001		MA	yes				
15.06.09	test_active_8.xls	10	1000	1001		CS	-				
16.06.09	test_active_9.xls	10	1000	1001		MA	yes				
16.06.09	test_active_9.xls	10	1000	1001		CS	-				
16.06.09	test_active_10.xls	10	1000	1001		MA	yes				
16.06.09	test_active_10.xls	10	1000	1001		CS	-				
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes				
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-				
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes				
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-				
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes				
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-				
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes				
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-				
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes				
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-				
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes				
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	ClientTrain Epochs	BackendTrain Epochs	minDurationOf ControlMillis	backendGender Param
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes				
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-				
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes				
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-				
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes				
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-				
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes				
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	CnetTrainbase name	BnetTrainbase name	Sw version	recv. Delegations backend
11.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	3
11.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
12.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	3
13.03.09	*pat_B_default_large.xls*	*10*	*1000*	*2001*	*2*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
16.03.09	*pat_B_default_large.xls*	*10*	*1000*	*1001*	*1*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	3
16.03.09	*pat_B_default_large.xls*	*10*	*1000*	*1001*	*1*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
17.03.09	*pat_B_default_large.xls*	*10*	*1000*	*101*	*0,1*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	
17.03.09	*pat_B_default_large.xls*	*10*	*1000*	*101*	*0,1*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	*301*	*0,3*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	1
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	301	*0,3*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	2
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	MA	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	2
18.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	CS	*yes*	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	0
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	101	*0,1*	MA	no	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	1
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	301	*0,3*	MA	no	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	1
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	501	*0,5*	MA	no	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	1
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	701	*0,7*	MA	no	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	1
19.03.09	*pat_B_default_large.xls*	*10*	*1000*	1001	*1*	MA	no	*train_mobile_default.xls*	*train_backend_default.xls*	*V1*	1
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	701	*0,7*	MA	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	501	*0,5*	MA	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	301	*0,3*	MA	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	101	*0,1*	MA	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	101	*0,1*	CS	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	301	*0,3*	CS	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	501	*0,5*	CS	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
23.03.09	*pat_B_default_large.xls*	*3*	*1000*	701	*0,7*	CS	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	CS	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0

date	database_name	Dataset width	Data base size	datasets process ed	Numb er of loops	mode	State caching	CnetTrainbase name	BnetTrainbase name	Sw version	recv. Delegations backend
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	2
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	701	*0,7*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	2
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	501	*0,5*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
25.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
25.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
26.03.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	yes	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	2
27.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	yes	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	2
27.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	yes	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	yes	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	yes	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
31.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
31.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
01.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	1
02.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	CS	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	0
02.04.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	CS	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	0
02.04.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	CS	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	0
06.04.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	CS	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	0
06.04.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	CS	no	*train_mobile_default_w7.xls*	*train_backend_default.xls*	*V1*	0
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	CS	no	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	1
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	CS	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	MA	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	CS	yes	*tests/traindata1.csv*	*train_backend_default.xls*	*V1*	0
26.05.09	test_active_1.xls	10	1000	1001		MA	yes	*train_mobile_default.xls*	*train_backend_default.xls*	V1	1
27.05.09	test_active_1.xls	10	1000	1001		CS	-				
27.05.09	test_active_2.xls	10	1000	1001		MA	yes				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	CnetTrainbase name	BnetTrainbase name	Sw version	recv. Delegations backend
27.05.09	test_active_2.xls	10	1000	1001		CS	-				
03.06.09	test_active_3.xls	10	1000	1001		MA	yes				
03.06.09	test_active_3.xls	10	1000	1001		CS	-				
05.06.09	test_active_4.xls	10	1000	1001		MA	yes				
08.06.09	test_active_4.xls	10	1000	1001		CS	-				
08.06.09	test_active_5.xls	10	1000	1001		MA	yes				
08.06.09	test_active_5.xls	10	1000	1001		CS	-				
09.06.09	test_active_6.xls	10	1000	1001		MA	yes				
09.06.09	test_active_6.xls	10	1000	1001		CS	-				
12.06.09	test_active_7.xls	10	1000	1001		MA	yes				
12.06.09	test_active_7.xls	10	1000	1001		CS	-				
15.06.09	test_active_8.xls	10	1000	1001		MA	yes				
15.06.09	test_active_8.xls	10	1000	1001		CS	-				
16.06.09	test_active_9.xls	10	1000	1001		MA	yes				
16.06.09	test_active_9.xls	10	1000	1001		CS	-				
16.06.09	test_active_10.xls	10	1000	1001		MA	yes				
16.06.09	test_active_10.xls	10	1000	1001		CS	-				
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes				
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-				
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes				
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-				
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes				
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-				
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes				
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-				
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes				
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-				
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes				
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	CnetTrainbase name	BnetTrainbase name	Sw version	recv. Delegations backend
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes				
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-				
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes				
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-				
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes				
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-				
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes				
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-				

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	client learning steps backend
11.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	39
11.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	0
12.03.09	pat_B_default_large.xls	10	1000	2001	2	MA	yes	39
13.03.09	pat_B_default_large.xls	10	1000	2001	2	CS	yes	0
16.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	yes	39
16.03.09	pat_B_default_large.xls	10	1000	1001	1	CS	yes	0
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	yes	
17.03.09	pat_B_default_large.xls	10	1000	101	0,1	CS	yes	0
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	yes	3
18.03.09	pat_B_default_large.xls	10	1000	301	0,3	CS	yes	0
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	yes	4
18.03.09	pat_B_default_large.xls	10	1000	501	0,5	CS	yes	0
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	yes	37
18.03.09	pat_B_default_large.xls	10	1000	701	0,7	CS	yes	0
19.03.09	pat_B_default_large.xls	10	1000	101	0,1	MA	no	35
19.03.09	pat_B_default_large.xls	10	1000	301	0,3	MA	no	84
19.03.09	pat_B_default_large.xls	10	1000	501	0,5	MA	no	130
19.03.09	pat_B_default_large.xls	10	1000	701	0,7	MA	no	207
19.03.09	pat_B_default_large.xls	10	1000	1001	1	MA	no	281
23.03.09	pat_B_default_large.xls	3	1000	1001	1	MA	no	178
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	MA	no	105
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	MA	no	52
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	MA	no	33
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	MA	no	21
24.03.09	pat_B_default_large.xls	3	1000	101	0,1	CS	no	0
23.03.09	pat_B_default_large.xls	3	1000	301	0,3	CS	no	0
23.03.09	pat_B_default_large.xls	3	1000	501	0,5	CS	no	0
23.03.09	pat_B_default_large.xls	3	1000	701	0,7	CS	no	0
24.03.09	pat_B_default_large.xls	3	1000	1001	1	CS	no	0

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	client learning steps backend
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	1001	*1*	MA	yes	14
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	701	*0,7*	MA	yes	14
24.03.09	*pat_B_default_large.xls*	*3*	*1000*	501	*0,5*	MA	yes	13
25.03.09	*pat_B_default_large.xls*	3	*1000*	301	*0,3*	MA	yes	13
25.03.09	*pat_B_default_large.xls*	3	*1000*	101	*0,1*	MA	yes	13
26.03.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	yes	38
27.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	yes	38
27.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	yes	18
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	yes	18
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	yes	18
31.03.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	MA	no	33
31.03.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	MA	no	60
31.03.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	MA	no	91
31.03.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	MA	no	166
01.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	MA	no	258
02.04.09	*pat_B_default_large.xls*	7	*1000*	1001	*1*	CS	no	0
02.04.09	*pat_B_default_large.xls*	7	*1000*	701	*0,7*	CS	no	0
02.04.09	*pat_B_default_large.xls*	7	*1000*	501	*0,5*	CS	no	0
06.04.09	*pat_B_default_large.xls*	7	*1000*	301	*0,3*	CS	no	0
06.04.09	*pat_B_default_large.xls*	7	*1000*	101	*0,1*	CS	no	0
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	MA	yes	16
06.04.09	*pat_B_default_small.xls*	10	300	1001	*3,34*	CS	no	0
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	MA	yes	15
08.04.09	*pat_C_default_small.xls*	10	300	1001	*3,34*	CS	yes	0
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	MA	yes	0
09.04.09	*pat_D_default_small.xls*	10	300	1001	*3,34*	CS	yes	0
26.05.09	test_active_1.xls	10	1000	1001		MA	yes	40
27.05.09	test_active_1.xls	10	1000	1001		CS	-	
27.05.09	test_active_2.xls	10	1000	1001		MA	yes	

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	client learning steps	backend
27.05.09	test_active_2.xls	10	1000	1001		CS	-		
03.06.09	test_active_3.xls	10	1000	1001		MA	yes		
03.06.09	test_active_3.xls	10	1000	1001		CS	-		
05.06.09	test_active_4.xls	10	1000	1001		MA	yes		
08.06.09	test_active_4.xls	10	1000	1001		CS	-		
08.06.09	test_active_5.xls	10	1000	1001		MA	yes		
08.06.09	test_active_5.xls	10	1000	1001		CS	-		
09.06.09	test_active_6.xls	10	1000	1001		MA	yes		
09.06.09	test_active_6.xls	10	1000	1001		CS	-		
12.06.09	test_active_7.xls	10	1000	1001		MA	yes		
12.06.09	test_active_7.xls	10	1000	1001		CS	-		
15.06.09	test_active_8.xls	10	1000	1001		MA	yes		
15.06.09	test_active_8.xls	10	1000	1001		CS	-		
16.06.09	test_active_9.xls	10	1000	1001		MA	yes		
16.06.09	test_active_9.xls	10	1000	1001		CS	-		
16.06.09	test_active_10.xls	10	1000	1001		MA	yes		
16.06.09	test_active_10.xls	10	1000	1001		CS	-		
19.06.09	test_actesc_300_50_1.xls	10	1000	1001		MA	yes		
22.06.09	test_actesc_300_50_1.xls	10	1000	1001		CS	-		
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		MA	yes		
22.06.09	test_actesc_300_50_2.xls	10	1000	1001		CS	-		
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		MA	yes		
23.06.09	test_actesc_300_50_3.xls	10	1000	1001		CS	-		
24.06.09	test_actesc_300_50_4.xls	10	1000	1001		MA	yes		
25.06.09	test_actesc_300_50_4.xls	10	1000	1001		CS	-		
25.06.09	test_actesc_300_50_5.xls	10	1000	1001		MA	yes		
26.06.09	test_actesc_300_50_5.xls	10	1000	1001		CS	-		
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		MA	yes		
29.06.09	test_actesc_300_50_6.xls	10	1000	1001		CS	-		

date	database_name	Dataset width	Data base size	datasets processed	Number of loops	mode	State caching	client steps	learning backend
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		MA	yes		
30.06.09	test_actesc_300_50_7.xls	10	1000	1001		CS	-		
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		MA	yes		
30.06.09	test_actesc_300_50_8.xls	10	1000	1001		CS	-		
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		MA	yes		
01.07.09	test_actesc_300_50_9.xls	10	1000	1001		CS	-		
01.07.09	test_actesc_300_50_10.xls	10	1000	1001		MA	yes		
02.07.09	test_actesc_300_50_10.xls	10	1000	1001		CS	-		

Database Name	Description
pat_B_default_large.xls	200 normal unsorted;100 active unsorted; 100 normal unsorted; 100 active sorted body_temp; 50 escalate sorted body_temp; 50 active unsorted; 100 norm unsorted;100 active unsorted; 100 escalate sorted by pulse; 100 active unsorted; 1000 records; N;A;N;A;E;A;N;A;E;A
pat_B_default_small.xls	50 normal unsorted; 50 active unsorted; 50 esc unsorted; 50 active unsort; 50 esc unsort; 50 normal unsorted 300 records; N;A;E;A;E;N
pat_C_default_small.xls	pulse fluctuation changed from 30% to 35%; 200 active records unsorted;50 esc unsorted; 50 active unsorted; 300 records; A;E;A
pat_D_default_small.xls	pulse fluctuation changed from 30% to 39%; body temp from 8 to 10%; 300 active records unsorted; 300 records; A
test_active_x.xls	1000 records, mainly active with manually inserted escalation records; Unsorted
test_actesc_300_50_x.xls	1000 records, mainly active with 50 generated escalation records every 300 records (new data generator feature); unsorted

Version	Description – Diff to prev Version
V1	Base Implementation. Client state caching, MA and CS mode, RLE-based xfer compression; Mobile Net is Trained on Backend, then deployed onto mobile. Mobile Evolving Net Params: 0,7/0,7/0,5/0,5; Backend performs a simulated(pseudo) high-sophisticated calculation and classifies the result with client net outputs to override possible client net Fps. Currently this is indicated by the backend net „gender" param -in MALE mode, the highlysophisticatedCalc. Assumes possible alarms are Fps because males are physiological fit, in FEMALE mode the backend net would not override the client nets alarms. Currently it makes no difference because the thing to be investigated is the systems behaviour when client identifies a FP,indifferent whether overriden by medical personnell on the backend office or by the backend classifier itself.

Appendix B

Input Data Generator Algorithm

The developed data generator tool (see 6.2.3) that was responsible for providing the simulated physiological test data for the experiments. The main algorithm worked as follows:

```
public void generateFile(final String filename) {
Map<DatasetElement, Double> prevValues =
        new HashMap<DatasetElement, Double>();
HSSFWorkbook wb = new HSSFWorkbook();
HSSFSheet sheet = wb.createSheet("data");
int actrow = 0;
actrow = generateHeaderRow(sheet, actrow);
for (int i = 0; i < numrecs; i++) {
        int cellnum = 0;
        final HSSFRow row = sheet.createRow((short) actrow++);
        for (DatasetElement e : def.elements()) {
                final Double prevVal = prevValues.get(e);
                final double val
                        = generateRandomValue(e, prevVal, i + 1);
                prevValues.put(e, Double.valueOf(val));
                HSSFCell cell = row.createCell(cellnum++);
                cell.setCellValue(val); }}
save(filename, wb);
```

The “generateRandomValue” method that is visible in the above main algorithm can be seen below:

```
private double generateRandomValue
(DatasetElement e, Double prevVal,int valueCount) {

//PH 20090618: handle auto-escalation
boolean autoEscalate = false;
if ((def.autoEscalateAfter != 0)
        && (valueCount > def.autoEscalateAfter)) {

        int rest = valueCount % def.autoEscalateAfter;
        autoEscalate = (rest <= def.autoEscalateFor);
        if (autoEscalate) {
                System.out.println("AUTO-ESCALATING␣record␣"
                + valueCount);
        }
}
//get a random value
final double rndPart = rnd.nextDouble();
//define boundaries
final double min = e.min;
final double max = e.max;
final double norm = e.norm;
final double fluc = e.fluctuation;
final double range_full = max - min;

//range in fluctuation of norm in percent:
final double min_fluc
        = (norm - ((range_full) * fluc));
final double max_fluc
        = (norm + ((range_full) * fluc));
final double range_fluc
        = max_fluc - min_fluc;

/**
* norm=50
* min=0
* max=100
*
* fluc=0,2
* 20% = 20
```

```
*
* min_fluc=30, max_fluc=70, range_fluc  = 40
*/
double retval = 0;

//generate a random value by mode
if ("normal".equals(mode)
        && !autoEscalate) {

         // 0 − 1 * range ==> value betwteen 0....40
        final double rndInRange
                = rndPart * range_fluc;
        boolean lastBelow = false;
        if (prevVal != null) {
                final double prev
                = prevVal.doubleValue();
                lastBelow = (prev < norm);
        }
        //jump over and under normal
        //value in "normal" operating mode
        final double sign = lastBelow ? 1 : −1;
        retval = norm + (sign * (rndInRange / 2));
}
else
if ("active".equals(mode)
        && !autoEscalate) {

        //no escalation no emergency,
        //but "active" patient
        //- i.e. making sports,
        //stepping stairs, bicicling etc.
        // use doubled normal fluctuation
        final double rndInRange
        = rndPart * range_fluc * 2;
        //tend to more activity
        final double sign
        = e.increaseOnActivity ? 1 : -1;
        retval = norm + (sign * (rndInRange / 2));
} else
if ("escalate".equals(mode) || autoEscalate) {
        //tend to more activity
        final double sign =
                e.increaseOnEscalation ? 1 : -1;
        //use normal fluc as base value
        final double rndInRangeNormal
                = rndPart * range_fluc;
        final double rndInRange
                = rndPart * range_full;
        // escalate exceed ranges
        final double rndInRangeTotal
```

```
                = rndInRangeNormal + rndInRange;
        retval = norm + (sign *
                (rndInRangeTotal / 2));
}
//handle extreme values that exceed limits:
if (retval < min)
{
        retval = min;
}
else if (retval > max)
{
        retval = max;
}

//hanlde bool values i.e. only min
//and max are allowed
if (e.isbool) {
        double dist2min = retval - min;
        double dist2max = max - retval;
        if (Math.min(dist2min, dist2max)
                == dist2min) {

                //generated random value is nearer
                // to min - return min
                retval = min;
        } else {
                retval = max;
        }
}
else if (e.isint) {
        return (int) Math.round(retval);
}
else {
        return retval;
}

}
```

Appendix C

Exemplary Generated Input Data

A representative subset of the generated input datasets for the simulation runs in the experiments is shown below. As described in 6.2.3, the data was generated by a specifically developed dataset generator:

Pulse	Body_temperature	blood_oxygenation	skin_resistivity_phase	skin_resistivity_amplitude	fall_sensor	GPS_latitude	GPS_longitude	GPS_height	environmental_data
56	36,25256445	6592	0,43854651	0,335671259	0	48,12146081	14,0844005	684	0
97	37,357161	7273	0,616215758	0,509321052	0	48,71535831	14,38951363	901	6
75	36,67584202	6479	0,322134924	0,339441765	0	48,18774616	13,91767554	447	0
80	37,34411832	7322	0,630661417	0,668053007	0	48,44092116	14,35089544	915	6
62	36,5484078	6866	0,422584243	0,358429105	0	48,10806717	14,0997762	501	0
82	37,54827654	7210	0,584203353	0,564300943	0	48,48744943	14,44044702	725	5
63	36,80662045	6123	0,439237934	0,474151428	0	48,21719542	14,01065895	610	1
92	37,48579532	7261	0,675826719	0,57855058	0	48,58302578	14,55781902	960	0
64	36,96839115	6930	0,33928984	0,310051558	0	48,18099333	13,8975181	655	4
97	37,27696548	7126	0,642058328	0,562840808	0	48,5571738	14,42008622	867	0
67	36,91230734	6250	0,396424054	0,419322797	0	48,332189	14,13973142	560	6
82	37,67411327	7227	0,50996121	0,558785192	0	48,36477882	14,2856433	976	0
76	36,41815037	6895	0,446752009	0,369699522	0	47,97258437	14,05864199	666	4
104	37,75544293	7707	0,686802345	0,549576472	0	48,50049058	14,31249422	866	0
74	36,8615839	6794	0,404447824	0,324205607	0	48,02308102	14,06127508	402	6
98	37,03684627	7790	0,578465095	0,516610088	0	48,53960654	14,58703147	867	0
60	36,1518038	6194	0,327725301	0,482283144	0	48,3275172	14,19107386	487	5
96	37,15644292	7194	0,671060678	0,521558051	0	48,34807466	14,35666709	962	0
61	36,35945717	6175	0,342791978	0,429903872	0	48,0784192	14,18022139	428	2
98	37,3868481	7384	0,683825834	0,56616975	0	48,67339573	14,55944276	928	0
72	36,46315297	6185	0,319157998	0,450195208	0	48,03237937	14,05710169	699	8
95	37,70790041	7978	0,606276682	0,594771584	0	48,55984553	14,36601213	902	0
77	36,24640268	6630	0,344169647	0,367792611	0	48,21988261	14,25893592	496	6
87	37,02781576	7623	0,568721517	0,598382742	0	48,72557777	14,53415409	900	0
60	36,24284512	6056	0,367879597	0,340559605	0	48,32901252	14,0336152	515	6
94	37,41346804	7939	0,562618151	0,554559594	0	48,69580969	14,32324116	985	1
74	36,75612496	6003	0,481358129	0,371496539	0	48,25457767	13,88623501	581	0
98	37,56073169	7995	0,625894252	0,693526587	0	48,70543238	14,52585313	993	5
72	36,29181011	6269	0,352291772	0,391588741	0	48,25380793	13,93482124	580	0
106	37,24355561	7854	0,638231801	0,518846846	0	48,36602893	14,44599133	917	4
61	36,15224905	6085	0,371207946	0,361024587	0	48,13688014	14,08622177	434	0
83	37,44567213	7384	0,532449828	0,529830925	0	48,4364237	14,58811497	799	2
57	36,50128439	6518	0,459906473	0,351643524	0	48,14514174	13,92712225	692	1
88	37,66157933	7461	0,520502154	0,693077518	0	48,54252719	14,61848597	850	0
79	36,46425498	6377	0,460670926	0,407653865	0	48,23201047	14,14662886	433	4
96	37,33366922	7125	0,656239888	0,591985269	0	48,36138098	14,62007524	775	0
61	36,14973422	6933	0,360574966	0,334589815	0	48,16113738	13,96584531	454	4
97	37,29010862	7004	0,51197108	0,645770049	0	48,61594835	14,38454794	911	0
76	36,42296794	6856	0,309691125	0,438156939	0	48,17353011	14,0463557	520	6

Appendix D

Wavecomm Tech Docs

D.1 Open AT Bluetooth Application Note, Revision 002 December 2005

APPLICATION NOTE

BLUETOOTH Interface

Revision: 002
Date: December 2005

OPEN AT

wavecom
Make it wireless

Operating Systems | Integrated Development Environments | Plug-Ins | Wireless CPUs | Services