Project Identification	
Project number	AAL 2012-5-199
Duration	1stMay 2013– 30th April 2016
Coordinator	Martin Morandell
Coordinator Organisation	AIT Austrian Institute of Technology GmbH, Austria
Website	www.relaxedcare.eu

Analysis of possible target areas and suitable existing mathematical models

Document Identification	
Deliverable ID:	D4.1B Analysis of possible target areas and suitable existing mathematical models
Release number/date	V04 18.05.2016
Checked and released by	Jonathan Steinhart (AIT)
Work Status	**Finished**
Review Status	**Accepted**

Key Information from "Description of Work"	
Deliverable Description	**. Analysis of possible target areas and suitable existing mathematical models**
Dissemination Level	Public
Deliverable Type	**Report**
Original due date	30.04.2015
Authorship& Reviewer Information	
Editor	Jonathan Steinhart
Partners contributing	
Reviewed by	Jure Lampe, SEN

The project RelaxedCare is co-funded by the European AAL JP and the following national authorities and R&D programmes from Austria, Switzerland, Slovenia and Spain

Release History

Release Number	Date	Author(s)	Release description /changes made
V 01	24.04.2015	JDS/AIT	First version, criteria not yet finalized
V 02	7.05.2015	JDS/AIT	Final draft
V 03	12.05.2015	JDS/AIT	Revisions suggested by reviewer.
V 04	18.05.2016	JDS/AIT	Minor edits for partner updates, final release

Relaxed Care Consortium

Relaxed Care (AAL 2012-5-199.) is a project within the AAL Joint Programme Call 5
The consortium members are:

Partner 1	**AIT AUSTRIAN INSTITUTE OF TECHNOLOGY GmbH (AIT, Project Coordinator, AT)**
Contact person:	Martin Morandell
Email:	Martin.morandell@ait.ac.at

Partner 2	**Hochschule Luzern Technik & Architektur – iHomeLab (IHL, CH)**
Contact person:	Martin Biallas
Email:	Martin.Biallas@hslu.ch

Partner 3	**50plus GmbH (50p, AT)**
Contact person:	Cornelia Huis
Email:	Cornelia.huis@50plusgmbh.com

Partner 4	**New Design University (NDU, AT)**
Contact person:	Sandra Dittenberger
Email:	sandra.dittenberger@ndu.ac.at

Partner 5	**SenLab (MOB, SI)**
Contact person:	Jure Lampe
Email:	jure.lampe@senlab.io

Partner 6	**Szenographie (SZE, CH)**
Contact person:	Ralph Eichenberger
Email:	contact@szenografie.com

Partner 7	**Ibernex (IBE, ES)**
Contact person:	Antonio Remartinez
Email:	antonio.remartinez@ibernex.es

Partner 8	**Soultank AG (SOUL, CH)**
Contact person:	Bianca Redel
Email:	bianca.redel@soultank.ch

Partner 9	**Schweizerisches Rote Kreuz Luzern (SRK, CH)**
Contact person:	Beatrice Burch
Email:	Beatrice.burch@srk-luzern.ch

Table of Contents

Release History *1*
Relaxed Care Consortium *2*
Table of Contents *3*
Abbreviations *4*
Executive Summary *5*
1 About this Document *1*
1.1 Role of the deliverable 1
1.2 Relationship to other Relaxed Care deliverables 1
2 Model Criteria *2*
2.1 Interpretability of parameters 2
2.2 Support for all data types 2
2.3 Information fusion 2
2.4 Incorporation of prior domain knowledge 3
2.5 Quantification of uncertainty 3
2.6 Minimal post-install training requirements 3
2.7 Ability to incorporate add-ons without re-training 3
2.8 Difficulty of implementation 4
2.9 Complexity of configuration 4
2.10 Computational requirements 4
3 Models and Analysis *5*
3.1 Box-Jenkins, extensions 5
3.2 Boosted trees 6
3.3 Dynamic Bayes networks 7
4 Discussion and Conclusion *9*

Abbreviations

Abbrev.	***Description***
RC	RelaxedCare
BPR	Behavior Pattern Recognition
AP	Assisted Person
IC	Informal Caregiver
AAL	Ambient Assisted Living
DBN	Dynamic Bayes Network

Executive Summary

This document summarizes the final results and findings from the analysis phase of the behavior pattern recognition work package in RelaxedCare.

First, evaluation criteria are chosen to ensure alignment with RelaxedCare's needs and goals. These criteria cover areas such as mathematical properties, features which affect applicability to real-world phenomena, implementation details, and reusability by future researchers.

Then, three model classes which would be structurally capable of being used within RelaxedCare are described. Each is then considered with respect to each of these criteria. While the comparison is essentially qualitative, a score is assigned as a means of summarizing and comparing the pros and cons of each model.

Finally a rough plan is proposed to proceed with the chosen class of models, dynamic Bayesian networks, such that the project goals are met and the identified risks and weaknesses are mitigated.

1 About this Document

1.1 Role of the deliverable

From the description of work:

> Based on the experience and first results of the requirements analysis mathematical models and algorithms will be evaluated concerning their applicability. For evaluation a list of criteria, such as flexibility, amount of training data used, complexity of configuration, has to be defined.
>
> A special focus is put on “measuring social interaction” as this should be one of the innovative parts of the project.
>
> For further specification of the models a data analysis will be performed and the required abstraction level of the data for each module as to be analyzed.

1.2 Relationship to other Relaxed Care deliverables

The deliverable is related to the following Relaxed Care deliverables:

Deliv:	*Relation*
D4.1A	This was a precursor to the present document and included background on the requirements for pattern recognition within RelaxedCare as well as a general taxonomy of models and algorithms which might be applied in BPR. It is assumed that the reader has at least read sections 2 and 3 from that document. Furthermore, whereas that document focused more on target patterns, the present one takes the targets as a given and focuses on model choice.

2 Model Criteria

In this section, we present the criteria which will be used to evaluate each of the candidate model families. In general, we are interested in models with properties that make them not only applicable to RelaxedCare's goals, but also easily generalizable for future AAL projects.

The BPR goals for the second prototype are (as with the first prototype) centered around the following three areas:

Social interaction

The most important BPR goal is the estimation of APs' level of social interaction. Within the scope of the second prototype, this is defined to be the presence of more than one person in the area covered by the system, speech, or telephone calls. The former should be detected via patterns learned from labeled training data gathered in the lab, whereas the latter can be found deterministically via SenLab sensors. Relevant labels for this will include number of people in the room and speech of activity of user or another person, gathered via video and audio recordings in laboratory.

Activity

Another important goal is to detect the AP's overall activity level, especially within the context of their usual values (or deviations therefrom) during a typical day. These should also be identified via sensor traces, via patterns learned from training data. Labels for this will be created using the heart rate, breathing rate, and composite activity score provided by the Zephyr BioHarness 3.

Emotion

Unlike the other targeted patterns, this is included not due to explicit requirements, but simply because doing so does not add significant extra cost or burden and may (under certain models detailed below) lead to more accurate estimates of other, targeted variables. Again, this should also be found via patterns in sensor traces, based on relationships found in training data.Labels will come from the PANAS-X assessment, administered periodically during laboratory data collection.

Of particular interest is the extent to which it is possible to pick up reliable traces of these phenomena, given only the relatively low-level sensor profile which is envisioned for a basic RC installation. As stated, in order to meet these goals it will be necessary to collect training data in the laboratory. These data should include all of the RelaxedCare sensors as well as the extra sources needed for labeling, ideally with 4-8 hours' worth of data on 10 people.

2.1 Interpretability of parameters

For the immediate needs of RelaxedCare, it is only required that the system be able to detect e.g. social interaction or activity given sensor data — not that that the parameters which govern this relationship have any intuitive interpretation. However, clearly such information might be of interest in future product designs. It would allow modelers, developers, and future AAL researchers insight into the mechanisms by which true states result in various sensor traces. Therefore, we will prefer models with interpretatable parameters.

2.2 Support for all data types

It is not clear a priori whether the aforementioned target patterns will be best modeled as discrete or continuous (or some mix of the two). For instance, mood may be best viewed as a continuum or it may instead as a series of "markers" for e.g. depression, agitation, etc. For this reason, it would be desirable to have algorithms which are capable of both regression and classification.

2.3 Information fusion

We consider the degree to which a model is capable of simultaneously combining and making use of all of the available information, both at the lower level (commonly referred to as *sensor fusion*) and at higher levels (more generally referred to as *information fusion*).

As alluded to in the previous section, our goal is to detect our targeted aspects of behavior as precisely as possible, given only the simplest set of sensors necessary. Because sensors are imperfect, their signals can be noisy (sometimes a motion sensor fires even if there's nobody in the room). And because the sensors we'll use are relatively simplistic, their signals are not rich enough to discern between similar true states (e.g. a binary motion sensor fires once even if there are one, two, or ten people in the room). Due to problems like these, it is desirable to use algorithms which are capable of jointly considering multiple data streams to isolate ("de-noise") and disambiguate ("de-alias") the true signals based on the traces present in multiple sensor streams. While all of the models listed below allow multiple inputs, certain algorithms have been recognized as having especially good sensor fusion properties.

In addition to sensor fusion, it would be ideal to have a model which is capable of fusing information also at higher levels. As a prerequisite for this, we consider that it is necessary to first have a joint model of the targeted patterns (i.e. as opposed to a separate model for each pattern).

2.4 Incorporation of prior domain knowledge

A further desirable characteristic for a model (and one related to Information fusion) is its ability to make use of prior knowledge about the structure of each sensor's data as well as the relationships between all of the variables in the system. These may include structural details about how the sensor inputs relate to each other (increasing Information fusion, as defined above), how the sensor inputs relate to the target variables, and even how the target variables relate to each other. By making use of information that is known the modeler about the real-world problem represented by the model, a generic, off-the-shelf tool can be improved to reduce the overall uncertainty in the targeted variables.

2.5 Quantification of uncertainty

One of RelaxedCare's guiding principles is non-obtrusiveness — both to the AP and the IC. An important aspect of this is only giving notifications when we are fairly certain that they're accurate. This requires some measure of certainty that, based on the observed data (and any prior assumptions), the resulting inferences are true. The system can then be tuned based on user feedback to find an acceptable threshold for notification.

Some models are built from the ground up out of probability models, making this trivial. Others have been successfully "retrofitted" with some notion of probability, although the probabilities they yield are only relative at best, and are not in practice well-calibrated to empirical observations. The remaining group of models could not be used to obtain probabilities without significant added effort.

2.6 Minimal post-install training requirements

Many algorithms require some form of training in order to yield meaningful results. In the extreme, this could mean per-user data which must be collected and labeled (by the user or an installer) for some interval which could span minutes, hours, or even days. Since this would severely detract from the simple, system-in-a-box concept of RelaxedCare, any such requirements must be kept to a minimum. In practice, one way to achieve this would be by employing algorithms capable of un- or semi-supervised learning (although we mostly exclude the former case from our model candidates, since our task at hand is complicated enough, even with some labeled data).

2.7 Ability to incorporate add-ons without re-training

RelaxedCare should be marketed as a modular system featuring one or more basic configurations which can then be extended with one or more add-ons, which could conceivably include additional sensors and tracked variables. While some of these may be anticipated during the development of the initial base unit, care must be taken to employ models which allow the add-on to plug in to a running system without disturbing the "inferential state" of the system. Ideally, it should even be possible to add new, previously unanticipated sensors, variables, and parameters into the live system, maintaining/updating current belief state and using it to set any new variables into a reasonable default state.

2.8 Difficulty of implementation

As in any project with limited time and resources, a line must at some point be drawn between features that will be included and those that may only be considered for a later iteration. In RelaxedCare, a more complex algorithmic development would mean less time available for configuring, enhancing, and fine-tuning models. It could also result in buggier and more opaque code unless great care is taken (which can also increase development time). For these reasons, algorithms which are simpler to implement correctly are preferred.

2.9 Complexity of configuration

Beyond the immediate BPR needs of RelaxedCare, this work package aims to provide a set of reusable modules for future AAL work. These will be most useful if models can be specified, run, and troubleshot in a simple and intuitive way. Ideally, a modeler with limited or even no experience in statistics or machine learning should be able to assemble rudimentary models.

Although this would also be somewhat helpful within the project, the relative effort saved by simplifying these things would likely be less than the effort needed to design and implement an intuitive interface, and so our evaluation focuses on the future use case.

2.10 Computational requirements

RelaxedCare Prototype B includes a low-powered, solid-state miniature PC upon which the HOMER platform and other necessary in-house software run. While it is technically possible to increase the computing power of this machine, doing so would increase cost, heat output, energy requirements, size, and potentially noise production. It is would also be possible to offload heavier computation to e.g. a cloud computing platform, but this add further requirements for the internet connection's uptime and bandwidth. Although it is entirely possible to increase available computing power somewhat arbitrarily in future iterations (using better integrated hardware and/or cloud computing), algorithms which are capable of running on the currently specified hardware are preferred.

3 Models and Analysis

We have selected three model classes which could make reasonable starting points for meeting RelaxedCare's behavior pattern recognition goals. Each model will be evaluated with respect to each of the criteria described above. A more quantitative approach to this comparison might involve simulation and/or validation on real data, but this is not feasible here, given time and resource constraints. Therefore, we opt for a mostly qualitative comparison, using a simple 3-point scale for each criterion as a very rough way of seeing which models stand out for the task at hand. For a given criterion, the set of available models is considered and those with particular strengths or weaknesses are given a +1 or -1 on that axis, respectively, while all others are given a 0. There are 10 criteria and thus scores may range from -10 to 10.

The fields of machine learning and statistics have many other models which might bear consideration here. We have winnowed these down to just the following, though, due to success shown with prior BPR and AAL application, existing experience and expertise within the team, and (especially) clear disqualification based on our selection criteria.

3.1 Box-Jenkins, extensions

The so-called Box-Jenkins approach to time series analysis is based on autoregressive, integrated, moving-average (ARIMA) models, with extensions to allow for vector responses, exogenous regressors, and seasonality (the latter of which could, in our case, be used to account for weekly and daily cycles).

- **Parameter interpretability(+1)**

Parameters in these models would be interpretable as changes in the expected target response, conditional on having made some particular sensor observation.

- **Discrete/continuous/categorical(-1)**

Response variables may only be real numbers.

- **Information fusion(-1)**

By jointly modeling the sensors' influences upon expected target values, and by allowing for interactions between them, it may be argued that a degree of Information fusion is allowed by ARIMA models. In practice, however, inclusion of many interactions or interactions of order higher than 2 or 3 can lead to numeric instability. It could also result in overfitting. Therefore, the modeler would likely have to manually choose a few of the many possible interactions to include, significantly limiting the degree to which fusion may considered to occur.

- **Incorporation of prior domain knowledge(-1)**

The only possible inputs from the modeler are which regressors to include. By choosing these to be pre-extracted features based on domain knowledge, this does allow some significant leeway. But the same can be done with any other model, giving the ARIMA no advantage here.

- **Quantification of uncertainty(+1)**

It is possible to perform hypothesis tests and produce confidence intervals for expected target values conditional on given sensor observations and parameters.

- **Post-installation training(0)**

It should be the case that some parameters (e.g. the effect of refrigerator-openings on estimated activity level) can be considered as generalizable beyond an individual. In this case, these parameters might be estimated during development and then considered fixed inside a running system. All other parameters, though, would likely need to be estimated in a supervised fashion.

- **Seamless add-ons(-1)**

Parameter estimates are conditional on the model and so, if the model changes, then the parameters may change significantly. Therefore, adding and removing sensors or otherwise changing the model would require any estimates to be recomputed.

- **Implementation difficulty(+1)**

Most programming languages have libraries capable of fitting these models (and they are fairly simple to implement from scratch, given good a linear algebra library).

- **Configuration difficulty(-1)**

Determining the degree of autoregression, differencing, and moving average, and model diagnostics and fine-tuning are somewhat nuanced procedures relying on graphical assessments of stationarity, autocorrelation, and residual patterns.

- **Computational requirements(+1)**

Computational requirements are minimal and model fitting is fast on any modern system with a linear algebra library.

Score: -1

3.2 Boosted trees

Decision trees work by recursively partitioning the feature space into regions with similar responses, and then estimating the response in each region with some aggregate of the responses found among training data in that region. As the number of partitions is increased, the model can fit the training data arbitrarily well — unchecked, this results in high variance, instability, and overfitting. By restricting the number of partitions, this excessive variance is avoided, but at the cost of increasing the model bias, leading to underfitting. At the extreme, we have decision stumps, with only one or two partitions, which may fit the data only slightly better than chance.

Boosting (e.g. AdaBoost) is a method for taking high-bias, weak learners like decision stumps and, by aggregating the results of many with a focus on previously-erroneous cases, reducing the bias without much increasing the variance, resulting in an overall learner with good accuracy on unseen data. It has been called the best "off-the-shelf" learning algorithm.

Unlike the other models, boosted trees were not developed with time series data in mind. The common practice for applying these to time series is to partition the training data into k-length windows and to produce a summary for each input and the response within this window. The model is then trained on these summary-input and summary-response pairs (i.e. one pair for each window). To account for the serial correlation which is characteristic of time series, the summary-response value for the last window (and maybe more past windows) is included as a feature during training. Then in the field, when the model is running with unlabeled data, the previous summary-response is estimated with the value predicted from the previous window.

- **Parameter interpretability(0)**

While the individual trees can be neatly interpreted as decision rules based on sensor values, this direct interpretability is mostly lost in the boosting process. However, there are some ways to see, in aggregate, which sensors tend to be more important, and what kind of influence the have.

- **Discrete/continuous/categorical(+1)**

These algorithms can work with all types of data as inputs and outputs.

- **Information fusion(0)**

Some prior research CITE has shown that AdaBoost has good sensor fusion properties. Intuitively, this makes sense: individual trees may be seen as rulesets which use inputs to disambiguate based on successive feature values, leading to more certain conclusions; and the entire boosted ensemble would retain some of this. Since these models only account for one output at a time, however, they cannot integrate information at any higher level.

- **Incorporation of prior domain knowledge(-1)**

The only choices the modeler has is which features to use, the parameters of the model (e.g. if using decision stumps, this means the boosting loss function, and the depth and impurity metric used to construct the underlying trees), and when to stop training.

- **Quantification of uncertainty(0)**

CITE tests methods for calibrating the probabilities produced by boosting, with good results. We are not aware of any method for obtaining probabilities from regression trees, however.

- **Post-installation training(-1)**

There is no obvious way to incorporate population-level parameters into this model.

- **Seamless add-ons(-1)**

There is also no obvious way to incorporate new sensors or other information into the model: changing the feature set can cause individual trees to change dramatically from root to leaves; and the progression of the sum depends very much on the performance of each subsequent tree. In short, changing anything would require re-training the entire model.

- **Implementation difficulty(+1)**

While implementation from scratch is relatively simple, several efficient and well-tested implementations already exist for the Java runtime.

- **Configuration difficulty(+1)**

Configuration is minimal.

- **Computational requirements(+1)**

These models are not costly to train (especially if the underlying trees are restricted to stumps), and running the model on unseen data is quite simple.

Score: 1

3.3 Dynamic Bayes networks

Bayes networks are an extremely broad and flexible class of models which are based on full probability models of domain variables. These can be depicted as directed, acyclic graphs and so variables may be connected hierarchically (with connections either being deterministic or stochastic), and data may be of any type. Due to their use of Bayesian probability, not only hidden states but also parameters have probability distributions. So we can estimate, for example, the probability that 2 or more people are present as well as the difference in probability of two people being present at noon vs. at midnight. The "dynamic" element referred to in the name refers to allowing past values of variables in the Bayes net to also have an influence on present values. The result of model estimation is a joint probability of unknown variables (including parameters) given all observed data.

Several of the most common and notable models usually employed in AAL (and in latent state tracking from sensor data in general) — e.g. hidden Markov models, state space models, latent Dirichlet allocation, generalized linear mixed models, mixture models — are special cases of dynamic Bayes networks (or minor variations thereupon).

- **Parameter interpretability(+1)**

Parameters specify the probabilistic relationships between domain variables and, as such, will often be of direct interest.

- **Discrete/continuous/categorical(+1)**

All types of data are supported.

- **Information fusion(+1)**

DBN's are a common sensor fusion method (as are special cases such as state space models and the Kalman filter in particular). The fact that all unknowns are jointly estimated via the model, incorporating (almost) arbitrary structure between all variables, leads us to suspect that DBN's may provide a high degree of information fusion at all levels.

- **Incorporation of prior domain knowledge(+1)**

Domain knowledge can be incorporated via choice of latent variables, graph structure, connecting distributions, and prior distributions.

- **Quantification of uncertainty(+1)**

DBN's are explicitly defined as joint probability models over all unknowns, and therefore can support arbitrary inference of these, jointly or marginally.

- **Post-installation training(+1)**

Population level parameters can be estimated in the laboratory and incorporated into per-user models. Furthermore, since all unknowns are also random variables, it is possible to e.g. marginalize the full joint distribution over the unknown user-level parameters to find the joint marginal distribution of the target states, given the data. This gives the possibility of eliminating post-installation training altogether.

- **Seamless add-ons(+1)**

Adding new sensors is fairly straightforward, requiring the model only to specify how the pre-existing variables in the model are expected to probabilistically affect the sensor. Other, more complex types of additions are also possible, when existing parameter estimates and structures can be retained.

- **Implementation difficulty(0)**

The complexity of a given DBN and the complexity of the algorithm needed to do practical inference on it are closely related. Some simpler (although still common and useful) models can be solved in closed form with tens of lines of code and suitable linear algebra and probability libraries. Many of the popular and more involved special cases (e.g, hidden Markov, hierarchical linear models, Kalman filters) have many efficient and well-tested implementations to choose from. To deal with more complex models or more general settings, though, more complex methods like Monte Carlo or variational methods become necessary. These are non-trivial to implement and test.

- **Configuration difficulty(0)**

Due to the graphical nature of the model and its composition from intuitive domain concepts, constructing the graph should (given a well-defined interface to the tools) be fairly straightforward. The issue of choosing probability distributions (including priors and hyperpriors) may present more difficulty, requiring some knowledge in this field or a well-designed interface for model building.

- **Computational requirements(0)**

The computational requirements mirror the progression from simple to complex in the above discussion of implementation. The simpler models have quite modest computational requirements, whereas the more potentially complex ones may require significant processing power (which in many cases can be parallelized) and RAM.

Score: 7

4 Discussion and Conclusion

Clearly, the dynamic Bayes networks bear consideration as our model class of choice. Our evaluation shows that they excel in most regards at meeting RelaxedCare's needs. However, it is essential that its shortcomings — namely, possible difficulty in implementation and configuration, as well as potentially high computational costs — are carefully considered, since they could still lead to critical problems if not mitigated. What follows are our recommendations for a path forward which properly protects against these contingencies. In general, we recommend a two-stage approach, corresponding to the two BPR goals of RelaxedCare: to solve RelaxedCare's problems, and to create reusable modules for others to solve their problems.

For the first phase, the task is essentially that of incorporating BPR functionality into the second prototype. Since all three of the aforementioned problems are directly related to the complexity of the models as realized, the decision must be made first what exactly the prototype can and will do with BPR. As training data become available, it will become possible to easily test and tune the performance of concrete models. There is a positive association between model complexity and richness of inference, but this levels off at a certain point of complexity. To get the richest inferences possible, we propose starting with simple models and iteratively increasing model complexity until the accuracy of the inferences drops off, or the point is reached where the ability to implement the models in the given timeframe is at risk, or the computational complexity reaches a point which cannot be accommodated on the given hardware within the given time. Then these models should be implemented, integrated into the prototype, and tested.

After work required for the protoype is complete the second phase can begin, where the task is to generalize these results. The focus must be on showing what is possible and a realistic means of reaching this. At the least, there should be a design which includes a language for specifying models, an engine which operates on these models, and an OSGi service which provides access to these. Time permitting, this design can be actualized as a series of interface and no-op implementations, and the solution chosen above (which might be a very specific case, for example, a hidden Markov model) can be re-framed within this framework as a first example. Finally, if time and resources remain, specific parts of the designed system can be more fully realized.

Since RelaxedCare aims to create a working, market-oriented prototype, it is important to ensure that some working and novel BPR functionality be included in the final prototype. Since RelaxedCare has a research component, though, it is also important that we explore the possibilities for what else could be done in this direction, and then share our findings and concrete results with the community. In fact, showing the potential in this way also increases the market appeal of the prototype.