Anomaly Detection in User Daily Patterns in Smart-Home Environment

Marek Novák, František Jakab, and Luis Lain

***Abstract* — The paper presents a technique for anomaly detection in user behavior in a smart-home environment. Presented technique can be used for a service that learns daily patterns of the user and proactively detects unusual situations. We have identified several drawbacks of previously presented models such as: just one type of anomaly - inactivity, intricate activity classification into hierarchy, detection only on a daily basis. Our novelty approach desists these weaknesses, provides additional information if the activity is unusually short/long, at unusual location. It is based on a semi-supervised clustering model that utilizes the neural network Self-Organizing Maps. The input to the system represents data primarily from presence sensors, however also other sensors with binary output may be used. The experimental study is realized on both synthetic data and areal database collected in our own smart-home installation for the period of two months.**

***Index Terms* — anomaly detection, behavioral patterns, clustering, Self-Organizing Maps, smart-home**

I. Introduction

ELDERLY people are often living alone, demanding increased attention of relatives or friends. Even one fall can lead to a dangerous situation as the loss of consciousness and mobility. If not controlled frequently, a person may lay on a ground for many hours or even days without any help. A prompt action may save the life. Smart-home systems, which are currently getting very popular, may act as detectors of such anomalous situations and therefore be supportive for relatives and carers. The benefit is on both sides, the elderlies would feel safer and the relatives would not be under such pressure to control the beneficiaries very frequently.

The functionality such as measuring values of sensors and actuating devices is already commercially available. These systems are marked with a term *domotics* and provide improved convenience, comfort, energy efficiency and security. At present, the focus is oriented more on processing of data gathered from ambient sensors. The aim is to make these systems more "intelligent", to allow them to learn the habits and patterns of the habitant adapt to them and operate in a proactive way.

M. Novak and F. Jakab are with the Department of Computers and Informatics, Technical University of Košice, Letná 9, Košice, Slovakia, e-mail: marek.novak@tuke.sk, frantisek.jakab@tuke.sk

L. Lain is with Tecnodiscap, University of Zaragoza, Campus Río Ebro C/Mariano Esquilor S/N Edificio I+D+I, Zaragoza, Spain, e-mail: llain@unizar.es

While working on a project *MonAMI* [1] we have built a service that was able to detect anomalous firings from presence sensors. Standard behavior of the user was not trained from observation (measurements from sensors), however was configured manually by an administrator based on a survey with the user. The users were asked to specify time when they wake up, go to sleep and leave the home. The service was aimed to send an alarm message when a measured value does not fit into a predefined interval. When filling the questionnaires, the beneficiaries were doubtful and the collected data did not serve us for the service configuration as we expected [2]. This is a significant drawback that prevents the successful usage of such a service.

In this paper we present a novelty method for a system which is capable to learn the behavioral patterns of the user by observing and to detect anomalies among recognized patterns. Detected anomalies may be used as an informative input for a carer, relative or a friend who can react promptly. The presented system utilizes neural network *Self-Organizing Maps* (SOM).

Section II presents related works in the area of anomaly detection in behavioral patterns. Section III describes types of anomalies our system is able to recognize, input values of the system and clustering approach. Section IV presents updated algorithm of the SOM for the purpose of our anomaly detection service. Section V. illustrates the model itself and required parameters. In section VI is shortly described a technical solution of our smart-home installation based on OSGi framework. Section VII discusses experimental study realized both on synthetic data and real database collected in the period of two months.

II. Related Study

Although there have been realized several works devoted to anomaly detection in human daily patterns in recent years, they are still far from real adoption and they suffer of many drawbacks. These works differ in sensing hardware and machine learning techniques used to detect the anomalies. Below are listed selected works that are very close to that of ours and which inspired us to propose a novelty method solving their deficiencies.

V. Guralnik and K.Z. Haigh [3] were among the first ones who have proven the possibility to find patterns in human behavior in a smart-home environment. They have proposed a machine learning approach to model human sequential patterns based on sequential pattern mining [4]. The fact, that

it is possible to identify human daily patterns, is an important assumption we built on.

J. Weisenberg et al. [5] created an inactivity threshold function. The main idea is to mark an event as anomalous when a monitored individual is inactive for an unusually long period of time based on historical data for a given time of the day. The threshold is a composition of maximum inactivity data point for a given time interval and two buffers to allow slight shifts in schedule. If detected inactivity exceeds the corresponding alert threshold, an alert is issued. A disadvantage of this approach is a rather long period of anomaly detection at night, so when the user loses consciousness at 2:00am, the system will recognize it only after 7 hours.

Another closely related work is of S. Mahmoud et al. [6], who compared binary similarity and dissimilarity measures for different days. They applied similarity measures as *Jaccard-Needham*, *Dice*, *Roger Tanmoto* and *Kulzinsky* on data gathered from occupancy sensors including door and motion sensors. As a main drawback of this approach we consider the inability to get more descriptive feedback about the anomaly. The result of their method is just binary, whether a particular day is significantly different (dissimilar) to any previous day, or not. It is not possible to acquire information which activity at which time was anomalous.

Similarly B. Kaluža and M. Gams [7] investigated the possibility to track changes in daily living dynamics. Firstly, they identified user's posture from 5 body-worn wireless tags with accelerometers. From the spatial coordinates, velocity and absolute distances between tags they classified one of the three postures: lying, sitting and standing. Secondly, they recorded for each particular activity a proportion of time the user performed it and monitored the changes in living habits. As a deficiency we consider the usage of obtrusive sensing (5 body tags) and anomaly detection only on a daily basis.

W. Kang et al. [8] considered activities as a hierarchical structure, where main actions (preparing breakfast, preparing dinner) are composed of sub-actions (sensor firings). They applied *hierarchical Hidden Markov Model* (HHMM) to find exceptional behavior patterns. A hierarchical topology of HHMM is mapped to the hierarchy of actions. Anomaly detection is based on time interval coverage of main actions to sub-actions. They assume sub-actions should last shorter than a covering main action. An important deficiency of this work is manual classification of sub-actions (sensor firings) into main actions.

Other works such as the one of O. Brdiczka, et al. [9] use various 3D video tracking, audio tracking systems or many specialized wearable devices to record and recognize specific postures and actions of the user. We consider such approaches obtrusive to users who refuse to use them because of the privacy loss. Similar response from users we get also from *MonAMI* project, where the users refused to use obtrusive sensors such as accelerometers attached to body and cameras. Therefore we have decided to focus on sensors in ambient environment, in particular presence sensors and reed switches.

Unlike the work of J. Weisenberg et al. [5] we focus also on other types of anomalies. In fact, our model can detect unusually long activity (inactivity), unusually short activity and unexpected activity at an unusual place and time. Thus we can cover also anomalies like falling down in the bathroom at night. Since we work with the knowledge of location, start time and duration of an activity, we can better describe when and where the anomaly occurred, in comparison to works of S. Mahmoud et al. [6] and B. Kaluža et al. [7], who just have the information that a particular day differs from previous days.

Our premises for the model are:

- to detect several types of anomalies, not just inactivity
- to use only unobtrusive sensors
- to avoid hierarchical level classification (main activities into sub-activities and sub-sub-activities)
- to provide early alert notification, not just on a daily basis

Simply put, our vision is to propose a model able to provide information where, when and what kind of anomaly happened to the beneficiary, as early as possible.

III. Related Study

We aim to design an unobtrusive outlier detection service based on motion (presence) data classification. Although we consider mainly the input from presence sensors, also reed switches and smart plugs may be added (a smart plug measures and monitors electricity usage of a plugged electric device). All these sensors provide binary output, if the device is in state ON or OFF.

The service trains for a specific time interval (e.g. one month) standard (expected) behavior of the user, given the observations. After this period it starts to evaluate observed activities and recognize whether they are anomalous or normal. Training is done by clustering observed data points into clusters of expected behavior. When the model is trained, the activities not fitting into any normal cluster are evaluated as anomalous. For this purpose we use an upgraded neural network *Self-Organizing Maps* (SOM).

We presume a person lives alone and pets are not considered, though this could be solved with a special RFID tag on a pet's collar and disregarding any sensor firings generated by a pet (if a pet is relatively small).

A. Types of Anomalies

The service can detect following anomalies:

- **unusual activity** (activity in unusual time) - An example situation of an unusual activity is when the user is sitting in the kitchen or the living room at night for some hours, but is expected to be sleeping. Wakeful nights are often a sign of some problem.
- **unusually long activity** - Unusually long duration of an activity may occur when the user falls or loses consciousness. Duration of such an anomaly will be unusually long. The model is not aimed to immediately safe the life (the fall will not be

immediately recognized), but rather provide an informative output if something unusual happens.

- **unusually short activity** - Unusually short duration of an activity may occur for instance at night, when the user wakes up much more early than usual, e.g. he is usually sleeping till 7:30am but one day wakes up at 2:40am. This may indicate health problems when the user is not sleeping peacefully.

B. Input Values

In our model we work with *activities*, which are the abstraction of real activities done by the user in a household environment. An activity could be: cooking, *sleeping*, *watching TV*, *bathing* and so on. The abstraction is based on our simplification of the real world, that an activity is composed of three values: *location*, *start time* and *duration*.

Let be an *i*-th activity a_i a tuple (1):

$$a_i = (l_i, s_i, d_i) \tag{1}$$

where:

- l_i is location of a sensor
- s_i is start time of an activity
- d_i is duration of an activity

Location l_i is usually a room, where the sensor is physically located, but not inevitably constrained to it. If an elderly would be living in a dwelling with rather big rooms, the surface may be separated in more locations.

Duration d_i in location l_i as a time interval between a time instant t_{start} when the state of the sensor is first changed from *OFF* to *ON* in location l_i, and time instant t_{end}, when the state of the sensor is first changed from *OFF* to *ON* in location l_j, where $l_i \neq l_j$.

C. Clustering

Given a set of all activities A we define a set $A_l \subset A$, which contains only activities from location l (2).

$$\forall a_i \in A_l : l_i = l \tag{2}$$

For each A_l we create standard behavior clusters C_l. Activities not belonging into C_l are considered anomalous.

Clustering algorithms from a set of *K-means* are not applicable in our model, due to the fact they require a number of clusters k in advance. For each location l there would be a different number of clusters and we never know them in advance. There could be only one cluster for a bedroom - sleeping at night, however three clusters for a kitchen - preparing breakfast, preparing lunch and preparing dinner.

Hierarchical clustering models like *Nearest-neighbor chain* or *CURE data clustering* are also not applicable because we are not building a hierarchy of activities in a particular location. Our objective is only to construct areas in a two dimensional space given by vector *(s, d)* of normal or expected behavior and the remaining part of this two dimensional space mark as anomalous.

For our model we have chosen the neural network *Self Organizing Maps*. SOM is usually used to visualize multidimensional data into 2 dimensional space [10] by grouping similar data together and thus reducing dimension of the input data set. The output is usually a two dimensional map with groups of different colors. Its advantage is that it does not require a number of clusters as an input parameter to the algorithm, preserve topological properties of the input space and moreover, the boundaries among clusters are not strict but rather gradient. The number of iterations influences how gradient is the boundary among clusters. The advantage is that it is configurable and we can tune them to our requirements.

IV. Upgraded SOM

Although an important aspect of the SOM is that it can classify data without supervision, we utilize it for semi-supervised learning. Semi-supervised in our context means, that SOM will not be trained on labeled training data consisting of vector pairs - an input vector and a target vector (both normal and anomaly data), but all training data would be considered normal. While testing the model, data far from normal regions would be considered anomalous.

As a result, we would like to get a learned SOM similar to that illustrated in Fig.1. On the left side there are gathered data for location *"Bedroom"*. On the right side is the trained SOM. Clusters of turquoise color going through orange to red represent normal behavior. In this example simulated data are directly mapped to SOM (indicated by arrows) and their pattern is preserved. Since this is not a standard SOM behavior, we have updated it slightly by changing initialization phase and weight vectors.

A. Standard Algorithm

Let x_k be the input vector of a dimension n (3) (upper index denotes an element in vector). Each neuron has its weight vector w_{ij} at position *(i, j)* where $1 < i < I$, $1 < j < J$ and I, J is the size of neurons lattice. Size of the weight vector is corresponding to the size of input vector of a dimension n (4).

Let t be an iteration index, λ be an iteration limit, α be a constant parameter for learning rate, u be the index of the best matching unit (BMU) described in the more detail below, σ_o be the starting radius around BMU given by I or J and $w(t)$ the weight vector in iteration t.

Description of an algorithm is taken from [10][11].

$$x_k = (x_k^1, x_k^2, \dots, x_k^n) : k \in \{1, \dots, N\} \tag{3}$$

$$w_{ij} = \left(w_{ij}^1, w_{ij}^2, \dots, w_{ij}^n\right) \tag{4}$$

1) Each node's weight is initialized to a random value from uniform distribution, usually in an interval *(0,1)* and iteration index t is initialized to *1*.

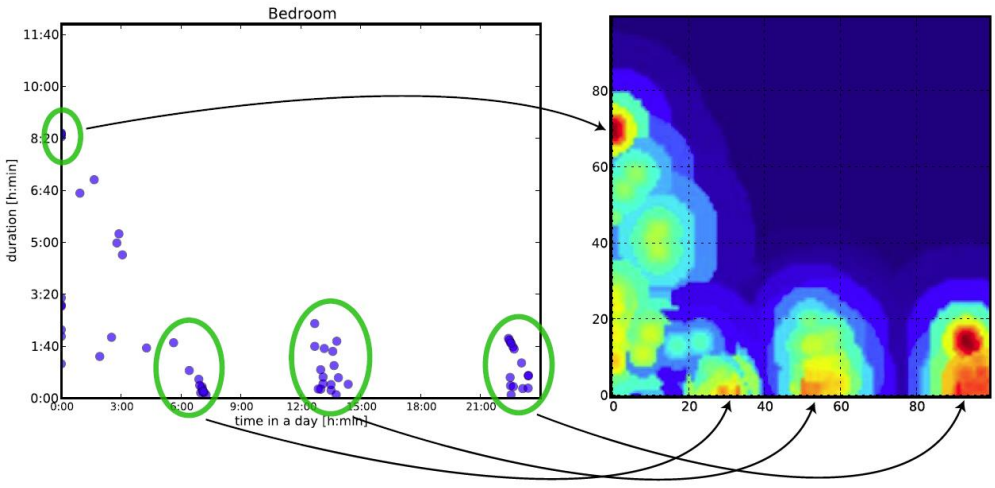

Fig. 1. An example how input data are transferred into learned SOM.

$$w_{ij} = U(0,1)$$

2) A vector x_k is chosen at random from the set of training data and presented to the lattice.
3) For each node there is calculated an Euclidean distance d_{ij} of the input vector x_k to weight vector of that node w_{ij}. Node with the minimum distance is called *Best Matching Unit* (BMU) and its position in the neurons lattice is defined as u. The weight of BMU is denoted as w_u.

$$\begin{gathered} \forall\, i,j:\ d_{ij} = d\left(x_k, w_{ij}\right) \\ d\left(x_k, w_{ij}\right) = \sqrt{\left(x_k^1 - w_{ij}^1\right)^2 + \cdots + \left(x_k^n - w_{ij}^n\right)^2} \\ u = \{i,j\}:\ \min_{ij} d(i,j) \end{gathered} \tag{5}$$

4) The radius around BMU given by a formula (6) is calculated for the iteration t. This area of the neighborhood around BMU shrinks in each iteration. A value σ_0 in this formula denotes a default size of the lattice in the iteration $t = 0$. If the lattice is a square, the $\sigma_0 = I/2$. If the lattice is not a square $I \neq J$, the $\sigma_0 = (\max(I,J))/2$.

$$\sigma(t) = \sigma_0 \cdot exp\left(-\frac{t}{\lambda}\right) \tag{6}$$

5) The learning rate value $l(t)$ given by a formula (7) is calculated for the iteration t. Similarly to the area radius, the learning rate decreases with each iteration. α is a parameter of the model.

$$l(t) = \alpha \cdot exp\left(-\frac{t}{\lambda}\right) \tag{7}$$

6) Every node within the BMU's neighborhood $\sigma(t)$, adjusts its weight vector by equation (8). The closer a node is to the BMU the more its weight is altered. $\theta(t)$ is a function decreasing with the distance (5) of currently processed node w_{ij} to BMU w_u (9).

$$\theta(t) = exp\left(-\frac{d(w_u, w_{ij})^2}{2\sigma^2(t)}\right) \tag{7}$$

7) Increment iteration index $t = t + 1$
8) Repeat steps 2 - 7 while $t < \lambda$

B. Weight Vectors

In standard SOM each point of the neuron lattice at position i,j is represented by a weight vector w_{ij}, where $w_{ij} = \left(w_{ij}^1, w_{ij}^2, \ldots, w_{ij}^n\right)$. When SOM is initialized the weights $w_{ij}^k; 1 \leq k \leq n$ are generated randomly and independently from an interval *(0,1)*. Given the input vector, the neurons weights are altered slightly to reduce the output error. This is iterated in a loop many times. Since weight vectors are initialized randomly, a trained SOM would for the same learning set look differently after each initialization. This means that recognized cluster may be located in the upper right part of the neurons lattice after first initialization, however in the lower left part of the lattice on the second initialization. We want to preserve the position and orientation of the clusters in the neurons lattice for the purpose of anomaly detection.

In upgraded SOM the weights are used only to identify the position of BMU in the lattice based on the input vector. Having a set of input vectors $x_k = (x_k^1, x_k^2); k \in \{1, \ldots, N\}$ and dimension of neuron lattice I,J we define initialization value for each neuron as (10).

$$w_{ij} = \left(\frac{\max\limits_{k=1,\cdots,N} x_k^1}{J} \cdot j, \frac{\max\limits_{k=1,\cdots,N} x_k^2}{I} \cdot i \right) \quad (10)$$

C. Semi-Supervised Learning

Our model is build on an assumption that learning set contains only normal data. For a selected time period, that is a learning period (e.g. one month), measured data are considered normal. Only after that period the model starts to detect outliers in human behavior. The outliers are data significantly far away from trained normal clusters.

At initialization time we would like to have a SOM, where the whole neuron lattice represents only one cluster - anomalous behavior. While learning the SOM, there are created clusters representing normal behavior. We never know in advance how many normal clusters the SOM will learn. This is in high contrast to the standard SOM, where weights are initialized randomly and therefore there are many clusters at the beginning and SOM iteratively converges to a number of few final clusters.

As we mentioned in the previous section, weights w_{ij} are used to find BMU in the lattice. While learning the model, weights' values are not amended and remain the same. Since weights w_{ij} are not used for learning, we define another weight value $v_{ij} \in (0,1)$ for each neuron in the lattice that is used for clustering.

Weights v_{ij} are initialized to *0*, where *0* represents anomaly behavior. While learning, weights v_{ij} are increased for inputs that represent standard behavior and converge to *1*.

Algorithm 1 Upgraded SOM algorithm

2: Set I, J, λ, α λ represents number of iterations
Set $\{x_1 \cdots x_N\}$
4: $max_{x_1} = \max_{k=1,\cdots,N} x_k^1$
$max_{x_2} = \max_{k=1,\cdots,N} x_k^2$
6: **for** $i := 1\ to\ I; j := 1\ to\ J$ **do**
$w_{ij}^1 = (max_{x_1} \cdot j)/J$
8: $w_{ij}^2 = (max_{x_2} \cdot i)/I$
end for
10: $t = 1$
while $t < \lambda$ **do**
12: **for** $k := 1\ to\ N$ **do**
for $i := 1\ to\ I; j := 1\ to\ J$ **do**
14: $d_{ij} = d(x_k, w_{ij})$
end for
16: $u = \{i, j\} = \min_{i,j} d_{ij}$
for $i := 1\ to\ I; j := 1\ to\ J$ **do**
18: $v_{ij}(t+1) = v_{ij}(t) + \theta(u,t,\alpha) l(u,t,\alpha)(x_k - v_{ij}(t))$
end for
20: **end for**
$t = t + 1$
22: **end while**

D. The Algorithm

The upgraded SOM is described in Algorithm 1. The main difference between standard and upgraded SOM is in initialization phase (lines 5 - 10), where w_{ij} is set based on the input set and amended weights v_{ij} are initialized to *0* for each neuron. Next important difference is located on line 18, where we update only weights v_{ij} for each iteration. Weights w_{ij} remain unchanged.

V. Anomaly Recognition

A. Parameters

During a day the user performs many unpredictable short transitions among the rooms, which are considered noisy and do not add any additional information to the main patterns which are in our focus. Therefore, we filter unwanted firings for activities shorter than a parameter τ_D, in particular, activities with duration $t < \tau_D$ are used neither in training phase nor for anomaly recognition. This parameter needs to be specified manually and reflects the period which is considered significantly important in anomaly recognition.

Showing it on the example with a kitchen and a TV, a difference between a time instant when the user goes into a living room to change a TV channel and a time instant when he comes back to the kitchen is relatively small, most probably less than a minute.

All parameters required by the model are the following:

- *I, J* – a size of the neurons lattice
- λ – a number of iterations used for learning
- α - a learning rate that specifies how quickly the BMU neighborhood shrinks over time
- τ_D - a time threshold that defines a filter for training data and recognition resolution

I represents duration of an activity and *J* represents time within the day (24 hours). If we set *I* and *J* to a number of seconds within the day, which is 86400, the neurons lattice would contain $\approx 2^{34}$ neurons. It would be computationally intensive and quite unnecessary. If we demand the precision of 5 minutes, as defined by τ_D, we can divide the day into 288 slots. The longest duration of an activity is sleeping and we are working with the value of 10 hours (since this parameter is configurable, we do not refer to any relevant study). When dividing 10 hours into 5 minutes intervals we get 120 slots, which is the value of *I*.

Apparently, the size of the neurons lattice I,J is dependent on the parameter τD:

- $I = maxduration/\tau_D$
- $J = 24/\tau_D$

Parameters α, λ have been identified experimentally by testing $\lambda \in \{10, 20, 50, 100\}$, $\alpha \in \{0.05, 0.2, 1.0, 5.0\}$. Based on the coverage of anomaly clusters we concluded to use $\lambda = 10$ and $\alpha = 5.0$.

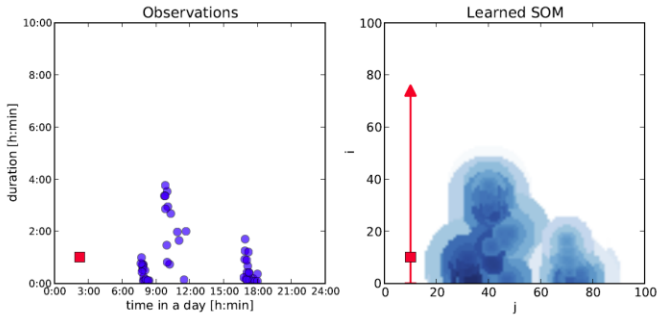

Fig. 2. Unusual activity.

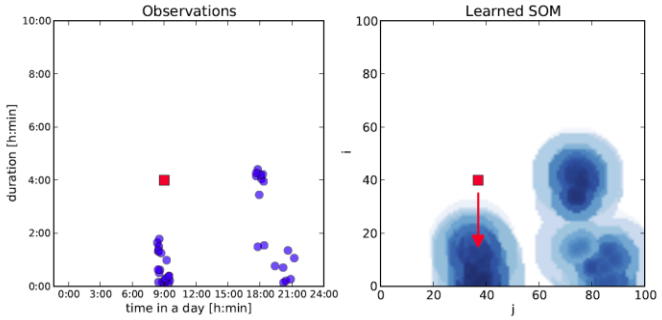

Fig. 3. Unusual long duration of an activity.

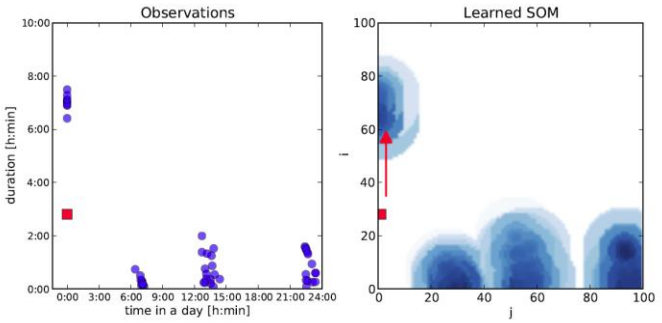

Fig. 4. Unusual short duration of an activity.

B. *Learning Phase*

Listing of Algorithm 2 represents a learning phase of the whole model. An input to the model is a set of locations L given by the arrangement of sensors in user's household and a set of activities A recorded during a learning period.

Firstly, there are created sets A_l containing just activities a_i from the location l (lines 7 - 13). Secondly, for each location $l \in L$ there is learned a SOM. Thus we get *normal clusters* C_l for every location l (line 16).

C. *Recognition Phase*

The anomaly recognition is realized in discrete time instants either when the user physically transits from one location to another (in the same manner as in the learning phase) or in time interval of τ_D with timestamp (synthetic) activities. When the user loses consciousness or falls and is not able to move, he does not perform any activities. Because of this, we use timestamp activities (with the location of last sensor firing) to identify if the current activity is not unusually long.

Algorithm 2 Model learning

```
2:  Set L, A                                   inputs to the model
    Set I, J, λ, α, τ_D                        parameters
4:  for i := 1 to L do
        l := l_i
6:      A_l := { }
    end for
8:  for i := 1 to N do
        a_i = (l_i, s_i, d_i)
10:     if d_i > τ_D then
            l := l_i
12:         A_l := A_l ∪ a_i
        end if
14: end for
    for i := 1 to L do
16:     l := l_i
        C_l = leanr(I, J, λ, α, A_l)           as defined in Algorithm 1
18:  end for
```

Anomalies are recognized in the following way (t denotes the current time instant):

- **unusual activity:** For this type of anomaly only real activities are considered. When such an activity does not belong to anomaly cluster in that particular location and is significantly long, it is considered anomalous (Fig. 2).
- **unusually long activity:** For this type of anomaly only timestamp activities are considered. Firstly the system needs to check whether for a particular location l_t and start time s_t there exists a normal cluster. Only if duration of the timestamp activity is longer than normal cluster (point (s_t, d_t) is above normal cluster) it is evaluated as an anomaly (Fig. 3).
- **unusually short activity:** For this type of anomaly only real activities are considered. A transit activity is evaluated anomalous if it does not belong into a normal cluster. The system checks whether for a particular start time s_t there exists a normal cluster, which duration is longer than that of tested a_t (Fig. 4).

VI. Smart-Home System Based on OSGi Platform

Real data for experimental verification of our model were gathered in *Valdespartera Living Lab* which is part of *European Network of Living Labs* (ENoLL). In the living lab there is installed a smart-home system implemented with several services for safe warnings, camera monitoring, automation, lighting scenarios, energy saving scenarios, daily tasks guiding, learning and traineeship, monitoring bio measurements, etc. [12].

The whole system is divided into three layers: hardware layer, core system and external services which may be located either on the same computer or may communicate remotely with the core infrastructure. The core system gathers and processes the data from hardware layer and decides upon configuration criteria whether values conform to a normal state of the household. If they do not (e.g. gas is detected) the

carer is informed about this situation through alerts such as SMS, email, call, or information on UI.

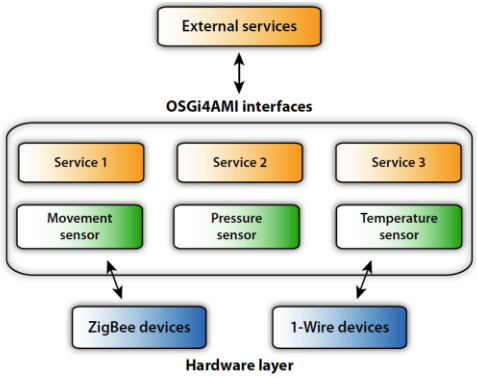

Fig. 5. Simplified scheme of osgi4ami-tecnodiscap middleware.

The core of the system is implemented in Java programming language, based on a modular framework *Apache Felix OSGi* [13]. Modules, in OSGi terminology, called bundles, interact through interfaces defined by *osgi4ami-tecnodiscap*.

A. OSGi4AMi Middleware

Ambient Assisted Living applications make use of the intermediate layer between device technology on one side and rule logic with user interfaces on the other. The advantage is the flexibility and adaptability of future hardware technologies as well as new scenarios based on beneficiaries' requirements. Such an approach helps also in development process, when several different developer groups may work on a common application in parallel.

Several middleware platforms have already been and still are in a development process in cooperative projects [14][15]. Although there have been various middleware proposed, none of them have become a standard yet. They are developed for slightly different domains and designed under different goals, therefore their implementation may vary. Though, the main idea is the same.

In our system we have adopted *osgi4mi-tecnodiscap* middleware [16]. This middleware is aimed at hiding hardware specific protocols and standards. For example, services do not have the information whether a temperature value came from a sensor built on a wired or a wireless technology. In Fig. 5 a simplified scheme of *osgi4mi-tecnodiscap* middleware is shown. The hardware layer (in our case ZigBee and 1-Wire network) is propagated through abstract interfaces as *TemperatureSensor*, *MotionSensor*, etc. to the layer of services. The interfaces are implemented in Java programming language and serve also as connection points for bundles in OSGi platform. The upper layer, labeled as *External services*, represents services or applications that are not a part of the OSGi platform.

B. System Overview

Global architecture overview of the system is depicted in Fig. 6. Residential gateway (RG) represents arbitrary computer with the support of Java Virtual Machine (JVM) and sufficient number of USB ports to connect device networks. There are located all OSGi bundles: a driver for each physical device technology, configuration bundle, logging support, rule management bundles and application server Virgo.

The user may interact with the system in different ways: by a smart-phone with Android operation system, through a multimedia center via a TV situated in user household or remotely by web user interface. The carer is notified about an unexpected situation by SMS and email alerts or directly by a phone call.

User interface provides a web UI for monitoring, controlling devices located in a household, calendar tasks management and some additional functionality such as weather forecast or interface for enabling and disabling system services.

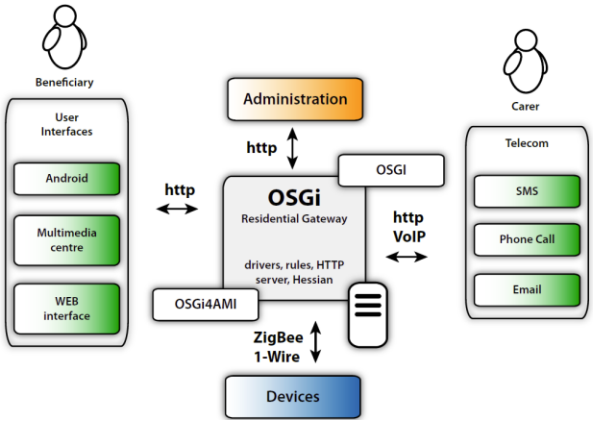

Fig. 6. Main components of the smart-home system.

C. Sensory Installation

In our experiment the monitoring was done in an apartment with one inhabitant, a PhD student, living alone for a period of two months. The apartment consisted of six rooms including kitchen and bathroom, however only four of them were really used. For the experimental study we have used only PIR (presence infra red) sensors. All the sensors were built on the ZigBee technology, developed at laboratory *CAITA-Tecnodiscap* (we have not used commercial ones). The placement of sensors is illustrated in Fig. 7.

The flat's area is divided into 5 locations: *Hall* - *Ha*, Kitchen - *Ki*, *Living room* - *Li*, *Bedroom* – *Be* and *Bathroom* - *Ba*. Both toilet and shower are located together in the bathroom.

Photos of installed sensors are presented in Fig. 8. Each sensory unit (small white box) is compound of three different sensors: a presence sensor, a light sensor and a temperature sensor. For the purpose of our research we have collected only presence sensor firings. As visible in the pictures, the sensors have wired power supply, albeit the data transmission is wireless. It is because the sensors required abruptly high energy consumption and the batteries were required to be changed every two weeks. This was a compromise we have made for the sake of higher quality data (to avoid missing data because of discharged batteries).

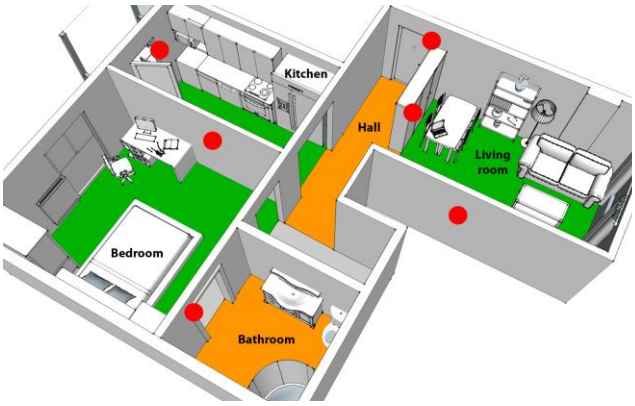

Fig. 7. Main components of the smart-home system.

VII. Experimental Results

To create a consistent and a suitable ADL dataset is a difficult task demanding reliable sensor network installation and sufficiently long time period for data acquisition. There are very few publicly available databases containing records from presence sensors collected in a smart-home environment. One of the publicly available databases is provided by MavHome project [17]. The problem is, they monitored a student with quite a hectic life and it is very problematic to find any patterns in his behavior. This is the reason, why we decided to build the simulator and to test the model on the synthetic data. We have also tested the model on data gathered in Valdespartera Living Lab presented in the previous section.

We presume a person lives alone and pets are not considered, though this could be solved with a special RFID tag on a pet's collar and disregarding any sensor firings generated by a pet (if a pet is relatively small).

Fig. 8. Placement of presence sensors within a laboratory flat.

A. Number of Training Days

Four SOMs learned for different learning periods (from 5 till 30 days) are presented in Fig. 9. The same as previously, the yellow color separates normal from anomaly clusters. As visible in the figure, there is not a big difference between a database consisting of 20 training days and a database consisting of 30 days (even 10 days seems to be sufficient).

To prove this assumption we have realized a test with different number of training days. The results are presented in Fig. 10, where the SOM was trained for 2, 4, 6, up to 40 days. The y-axis indicates a proportion of the area (in percentage) trained for normal behavior to the whole neurons lattice area (120 *x* 120 neurons). Above 16 training days the coverage area increases just slightly (even falls down for 22 days which is caused by the fact, that it is the probabilistic model). The increase from 20 training days (*13,8%*) up to 40 training days (*15,2%*) is only 1,4%.

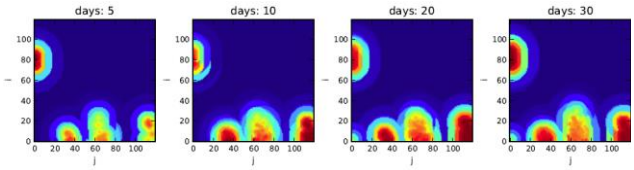

Fig. 9. SOMs learned on a simulated data using different training period, λ= 10; α = 5:0.

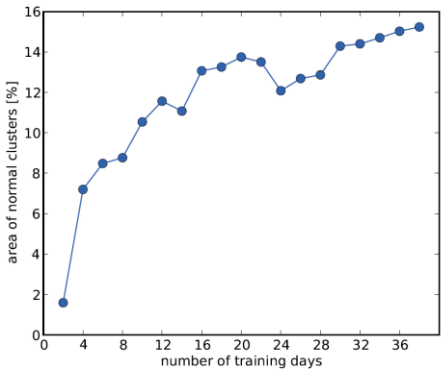

Fig. 10. The coverage area of normal clusters to the whole neurons lattice area for different training periods.

TABLE I
Confusion Matrix for Synthetic Data

Actual class	**Predicted class**		
	Normal	Unusual	Accur.
Normal	17 290	150	99.1%
Unusual	770	27 430	97.2%
Unusual long	1230	15 430	92.6%
Unusual short	940	4 860	83.8%

B. Results from Simulated Data

To experimentally verify the model we have used data from the simulator. The verification consisted of the following steps:

1) The simulator was used to generate a training dataset for a period of 20 days.
2) The model was trained on this dataset with parameters: λ = 10, α = 5.0.

3) Separate testing datasets were generated for each of: normal activities, unusual activities, unusually long activities, unusually short activities and activities in wrong location (for missing activity verification).
4) The confusion matrix was created for each testing dataset.

The simulator is built as a Markov process, where the probability of the next state is given only by the current state. Its objective is to simulate movement sensors - to generate a synthetic sequence of times t_i and locations l_i when presence sensor records movement of the user. While generating data, the simulator moves forward through the pattern states (these are the input to the system) and among these transitions generates noisy firings.

To generate a testing datasets we manually created reference points representing anomalies (e.g. firing at 4.00am in the bedroom). From these reference anomalies we generated synthetically data in a frequency of 5 minutes for both start time and duration of an activity. Thus we get hundreds of test cases. This was repeated for 10 different daily scenarios.

C. Results from Real Data

When processing the data, we needed to differentiate between weekends and working days. Weekends and holidays were a lot noisy without any regular patterns, because the student was on his research mobility and made trips or went to parties during his spare days. In the contrary, working days were quite regular with the majority of time spent at the university. Within the period of two months (60 days), there were 37 working days and 23 free days.

For training the model we used 20 days, the remaining days were used for testing. The trained SOMs are depicted in Fig. 11. The parameter τ_D was set to 5 minutes. As visible in the bedroom, the student used to go to sleep at *1:00am* and was sleeping from 5 to 8 hours. He never cooked by himself, therefore there are only short activities in the kitchen. A table, where the student used to eat, was located in the living room, which had also the impact on the duration of the activities performed in the kitchen. The mainly occupied location was the living room, where the student was working as well as relaxing (watching TV), which is visible on scattered points in evening hours. A daily hygiene he performed before going to bed (around *1:00am*) and in morning hours (around *7:00am*). Since the student was usually more than 10 hours out at the university and we wanted to preserve the parameters of the neurons lattice introduced before *(I = J = 120)*, the activities for location "out" were split into two parts - before *3:00pm* and after *3:00pm*.

At the time of data collection we did not have an implemented service for anomaly recognition. Because of this, we could test the model only afterwards by batch processing. We created normal activities from remaining days (20 days were used for testing and 17 for testing) and manually created anomalous scenarios.

The examples of anomalous scenarios are presented in Table II. Each entry corresponds to the one activity and provides the values: location, start time, end time, actual class, predicted class by the model and a short description when such an anomaly may occur. Daily pattern of the student was the same within the whole period of 37 days.

TABLE II
ANOMALOUS DATA USED FOR TESTING THE MODEL WITH A PREDICTED VALUE (U = UNUSUAL, L = UNUSUALLY LONG, S = UNUSUALLY SHORT)

Loc	Start	End	Actual	Predict.	Desc.
Be	9:40:00	14:30:00	U	U	unusual sleep
Ki	3:10:00	3:50:00	U	U	waken up
Ki	20:10:00	22:30:00	U	U	unusual
Li	2:20:00	3:40:00	U	U	waken up
Li	10:20:00	13:15:00	U	U	not at work
Ba	2:25:00	2:35:00	***U***	***Normal***	waken up
Ba	4:12:00	4:19:00	***U***	***Normal***	waken up
Be	0:40:00	9:30:00	L	L	long sleep
Be	2:05:00	10:00:00	L	L	long sleep
Ba	1:34:00	2:50:00	L	L	fall
Ki	7:34:00	9:30:00	L	L	fall
Ki	21:14:00	23:59:00	L	L	fall
Li	00:00:01	3:00:00	L	L	fall
Li	1:10:00	3:00:00	L	L	fall
Be	00:10:00	2:30:00	S	S	sleepless night
Be	00:50:00	4:10:00	S	S	sleepless night
Ou	7:50:00	10:50:00	S	S	not at work
Ou	7:50:00	13:00:00	S	S	not at work
Ou	15:00:00	16:10:00	S	S	early at home

For the anomalous activities we get the accuracy of *94.6%* (35 from 37 correct). Incorrectly were recognized short stays in the bedroom at night (in the table marked in bold and italics). It is caused by the fact, that SOM for the bedroom is trained for relatively wide range of start times, what is inappropriate (the normal cluster is too wide). For short activities (taking only 10 or 20 minutes) the shorter radius around BMU $\sigma(t)$ would be more suitable. This is the limitation of our model, which should be improved for precisely defined scenarios.

VIII. CONCLUSION

We have presented the service for anomaly detection in human daily patterns, that is capable to recognize anomalies like: unusual activity, unusually short activity, unusually long activity and missing activity (not present when expected) in data gathered from presence sensors. To our best knowledge, it is the first time a neural network *Self-Organizing Maps* is adopted for this purpose. Based on a shape of the input data (location, start time and activity duration), we have amended slightly the initialization and training phase of SOM. A separate SOM is created for every location.

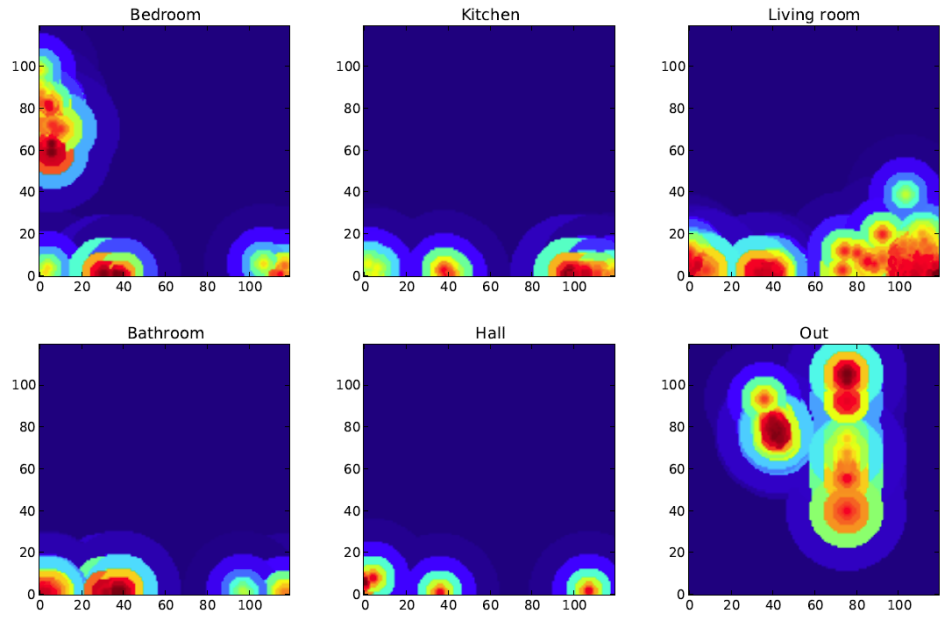

Fig. 11. Trained SOMs for real activities collected within the period of 20 days.

In comparison to other current approaches our service supports recognition of more anomaly types, provides more informative output what actually happened to the user and rather quick recognition response. Naturally, the recognition is not as quick as in fall detection systems, however the objective of our service is little different.

The experimental study was realized on synthetic firings from presence sensors and data gathered in real installation of a smart-home system as well. The model was tested on both datasets with promising results. The accuracy ranges from *83.8%* to *99.1%* for unusually short activities and unusual activities respectively. The model incorrectly recognizes anomalies which differ only in 10 or 20 minutes from standard behavior. Function $\sigma(t)$ should be amended slightly to better model around BMU.

We are aware of the fact, that rigorous testing requires a higher number of respondents and also a database containing data collected from a longer time period. Future research we see in real implementation and deployment of the model in several testing smart-homes and in cooperation with carers to optimize the parameters of the model.

ACKNOWLEDGMENT

This work is the result of the Project implementation: Competency Centre for Knowledge technologies applied in Innovation of Production Systems in Industry and Services, ITMS: 26220220155, supported by the Research & Development Operational Programme funded by the ERDF.

REFERENCES

[1] "Project page: Monami," may 2011, accessed: May 2013. [Online]. Available: http://www.monami.info/

[2] D. Šimšík, A. Galajdová, D. Siman, M. Novák, and P. Galajda, "Services for seniors - experience of testing in slovakia field trials," *Assistive Technology Research Series: Everyday Technology for Independence and Care - AAATE 2011*, vol. 29, pp. 1082–1089, 2011.

[3] V. Guralnik and K. Z. Haigh, "Learning models of human behaviour with sequential patterns," in *Proceedings of the AAAI-02 workshop "Automation as Caregiver"*, 2002, pp. 24–30, aAAI Technical Report WS-02-02.

[4] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proceedings of the Eleventh International Conference on data Engineering*, 1995, pp.3–14.

[5] J. Weisenberg, P. Cuddihy, and V. Rajiv, "Augmenting motion sensing to improve detection of periods of unusual inactivity," in *Proceedings of the 2nd International Workshop on Systems and Networking Support for Health Care and Assisted Living Environments*, ser. HealthNet 08, 2008, pp. 2:1–2:6.

[6] S. M. Mahmoud, A. Lotfi, and C. Langensiepen, "Behavioural pattern identification in a smart home using binary similarity and dissimilarity measures," in *Proceedings of the 2011 Seventh International Conference on Intelligent Environments*, ser. IE '11. IEEE Computer Society, 2011, pp. 55–60.

[7] B. Kaluža and M. Gams, "An approach to analysis of daily living dynamics," *World Congress on Engineering and Computer Science, WCECS 2010*, vol. 1, pp. 485–490, 2010.

[8] D. S. Wonjoon Kang, Dongkyoo Shin, "Detecting and predicting of abnormal behavior using hierarchical markov model in smart home network," in *Industrial Engineering and Engineering Management (IEEM)*, 2010, pp. 69–75.

[9] O. Brdiczka, M. Langet, J. Maisonnasse, and J. Crowley, "Detecting Human Behavior Models From Multimodal Observation in a Smart Home," *Automation Science and Engineering*, IEEE Transactions on, vol. 6, no. 4, pp. 588–597, 2009.

[10] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, Sep. 1990.

[11] M. Buckland, "Kohonen's self organizing feature maps," accessed: May 2013. [Online]. Available: http://www.ai-junkie.com/ann/som/som1.html

[12] "Valdespartera living lab," may 2013, accessed: May 2013. [Online]. Available: http://www.openlivinglabs.eu/livinglab/valdespartera-living-lab
[13] A. Felix, "Apache Felix - Overview," accessed: May 2013. [Online]. Available: http://felix.apache.org/
[14] M.-R. Tazari, F. Furfari, J.-P. L. Ramos, and E. Ferro, "The PERSONA service platform for AAL spaces," 2009.
[15] T. Fuxreiter, C. Mayer, S. Hanke, M. Gira, M. Sili, and J. Kropf, "A modular platform for event recognition in smart homes," in *e-Health Networking Applications and Services (Healthcom), 2010 12th IEEE International Conference on*, 2010, pp. 1–6.
[16] L. Lain, "osgi4ami-tecnodiscap," accessed: May 2013. [Online]. Available: http://sourceforge.net/projects/osgi4ami-tdc/
[17] G. Youngblood and D. Cook, "Data mining for hierarchical model creation," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 37, no. 4, pp. 561 –572, 2007.

Marek Novák is an internal PhD student at Technical University of Košice. His research interest is mainly in design and development of various services for a smart-homes environment. He has cooperated in an international project MonAMI. His PhD thesis is dedicated to finding anomalies in user behavioral daily patterns. Besides smart-homes, he is interested in e-learning supportive services and tools for programming courses. He is a member of Computer Networks Laboratory.

František Jakab received the MSc. degree in Systemotechnic engineering from the St. Petersburg Electrotechnical Institute (Russia) in 1984 and the PhD. degree in 2005. He is employed as an assistant professor at the Dept. of Computers and Informatics, Technical university of Košice, Slovakia. He is the head of the Computer Engineering Group and Computer Networks Laboratory. His research interests include projecting of computer network, modeling, simulation and network management, new form of multimedia-based communication, QoS, telelearning systems, intelligent tutoring systems. He has been a coordinator of several large international e-learning oriented projects supported by EC. He is a coordinator of the Cisco Networking Academy Program for the Slovak Republic and head of the Application Section of the Communication Technology Forum Association in Slovak Republic.

Luis Lain is a full-time researcher at CAITA-Tecnodiscap group at University of Zaragoza. He finished Computer Science studies in 2000 and has been working in main national and international IT companies (financial, telecommunications...) since then. Last 5 years he has participated in all the European and National projects the CAITA-Tecnodiscap group is involved in, leading technical coordination and managing resources and personnel (EU projects MonAMI, EasyLine+, etc.). His interests include Java developments, Open Source projects, ambient intelligence and assistive technology.