Published in The Computer Journal, Vol.53, No.2, pp.142-151, February 2010 (pre-publication version)
First published online *November 9, 2007*; doi: 10.1093/comjnl/bxm078

# Detection of Normal and Novel Behaviours in Ubiquitous Domestic Environments

F. RIVERA-ILLINGWORTH[1], V. CALLAGHAN[1], H. HAGRAS[1]

[1]*Computer Science Department, University of Essex, Wivenhoe Park CO4 3Q, United Kingdom*
*Email: friver@essex.ac.uk*

## Abstract

*The importance of ubiquitous environments has increased in recent years as it has been recognized as a paradigm that can improve the quality of life of many sectors of the population especially care of elderly people by providing automated environments that adapt and respond to its inhabitants' needs. The aim of the work presented here is to provide a solution to the problem of recognition and detection of human behaviours inside ubiquitous environments by using a neural-network driven embedded agent working with online, real-time data from a network of unobtrusive low-level sensors. The final objective of this system was to classify a "normal" pattern of activities, and sense deviations from it, which could be employed for home care applications.*

## 1. Introduction

Ubiquitous or pervasive computing relates to the concept first expressed by Mark Weiser, who had a vision of "People and environments augmented with computational resources that provide information and services when and where desired" [1].

Since this concept was first proposed in 1988, applications and ideas from many different fields such as computer science, electronics, psychology, sociology, architecture, etc. have converged in order to fulfil the notion of truly pervasive environments.

One of the areas of research related to pervasive environments is the recognition and detection of human activities inside an intelligent environment. By identifying patterns in behaviours a pervasive environment can respond and adapt itself to the inhabitants' needs more efficiently and in a personalized way, thus improving their quality of life.

An intelligent environment (e.g. a room, a house, an office, etc) can be described as a place that "has sensors, actuators and agent based control systems that monitor the occupants, communicate with each other, and intelligently control the environments to support the occupants in their daily activities" [18]. These environments can simply be automated systems or more complex adaptive systems that use some sort of "intelligence" in order to learn and adapt themselves to the inhabitant's behaviour. The applications of this kind of environments are very diverse, ranging from the automation of HVAC (Heating, Ventilation and Air-Conditioning) systems inside buildings, educational applications in classrooms, energy saving/management for offices or care at home systems, among others.

This paper presents an integral framework consisting of the development of both a simulated and two real environments monitored by a system using a neural-network-driven embedded agent able to work with online, real-time data from a network of unobtrusive low-level sensors. The objective of the system is to discover a normal habitual set of human activities and identify novelties in ongoing behaviour which do not fit the model of previously contructed behaviours or activities. Such novelties could, for example, relate to falls, or deteriorating medical conditions.

The remainder of this paper is divided as follows: Firstly, we present a brief introduction to the challenges and approaches involved in human activity detection. Secondly, we describe the experimental system comprising the environments, their sensors' architecture and the internal agent structure. We then present a description of the experiments, the results, followed by the conclusions and finally our future work plans.

## 2. Detection of human activities

One of the more important and useful metrics to study human behaviours inside a domestic environment, is tracking the ability of the people to perform the so-called Activities of Daily Living (ADLs) and the Instrumental Activities of Daily Living (IADLs) [9, 12, 13, 14, 16, 18].

ADLs focus on assessing the person's ability to perform basic self-care activities such as eating, dressing, bathing, going to the toilet and transferring in and out of bed/chair and walking. IADLs measure activities related to independent living and include preparing meals, shopping for personal items, medication management, managing money, using the telephone, laundry doing, and transportation ability. [13, 14]

Limited resources, especially in helping with ADLs encourage the development of technologies that can assist and support some sectors of the population, such as elderly people, their relatives and possibly the carers. ADLs depend on regular patterns of behaviour, and by learning such habitual patterns, it could be possible to recognise significant deviations from it and to infer possible problems or to provide appropriate assistance if needed.

It has been observed that sleeping disorders are common in people with dementia, and the sleep/wake rhythm in Alzheimer's disease is extremely disturbed. One of the warning signs of Alzheimer's disease is the difficulty to perform familiar tasks [22]. People with dementia often find it hard to complete everyday tasks that are so familiar that usually we do not think about how to do them. A person with Alzheimer's may not know the steps for preparing a meal, using a household appliance, or participating in a lifelong hobby or he/she might cook a meal but forget to serve it [23] .

There have been several projects aiming to apply the notion smart environments to the health care domain. Project such as Georgia Tech's Aware Home [], Smart Medical Home project at the University of Rochester [], Elite Care [], BT's Telecare [] , MIT'S PlaceLab [], among others, have been investigating the use of different technologies which provide automated data collection and recognition of human behaviours. For a more detailed account of the related projects and assistive technologies developed, please refer to [] and [].

The recognition of human activities poses several challenges due to the diverse number of ways in which people perform those activities, the configuration of a set of suitable sensors inside the environment and the architecture used to detect and classify the activities.

One of the areas that has been studied most intensely concerns the type of sensors and the way the can be configured in order to be used for activity detection purposes. It has been recognized that monitoring techniques that are relatively automated and unobtrusive are much more likely to be successful []. The consequence of using such techniques is that the acquired data is noisier, requiring more sophisticated algorithms for inferring the current state; however, such data will be collected in a continuous mode, without explicit user interaction which, in turn might be seen as compromising the user's privacy [6].

Some desirable characteristics for the sensor set are that it should be unobtrusive and that no modifications, or only minor modifications, to the environment are needed in order to deploy them [2]. The sensors also need to be reliable, need little, or no, maintenance and ideally be cheap, so they can be deployed in large quantities.

Among the current technologies used in smart environments are image recognition systems using cameras and visual sensors, the use of microphones, and even thermal imaging sensors [9, 16]. Although these approaches work well under laboratory conditions, they face many problems when used in everyday situations. Moreover, cameras are considered by many to be intrusive to people's privacy and most of the times, the interpretation of the images requires complex software.

The use of wearable devices such as accelerometers to detect physical activities such as walking, sitting, etc, or the use of biometric or emotion-detection sensors has been a common approach. A popular alternative is the use radio-frequency-identification devices (RFID) and tags which due to their low cost can be deployed in larger quantities. The drawback of wearable devices is that they rely on the users' cooperation to work properly.

The use of very simple on/off sensors such as motion detectors, pressure sensors, and switches has proved to be suitable to infer high-level behaviours from low-level sensory data. Some systems [12, 17] have already used this approach with good results, using cheap sensors that can be deployed at a low cost inside a home environment. For our system, we have chose to use a network of simple, non-obtrusive sensors like pressure sensors, motion sensors, light switches, etc., along with an embedded agent using a neural network in order to interpret the data. A complete description of the system will be presented in the following section.

## 3. System framework

The activity recognition system has 3 different elements: The set of sensors and their configuration, the agent that processes the sensor data, and the environment itself. This section describes these three parts using a top-bottom approach, beginning with a description of the environments, continuing with the sensors' configuration and finishing with the description of the architecture of the agent.

### 3.1. Experimental environments and sensor's architecture

In order to implement and test the proposed system, two real environments located at the University of Essex were used. The first one is called the intelligent dormitory (iDorm1). It is a testbed comprising a large number of embedded sensors, actuators, processors and networks in the form of a small self-contained room with areas for different activities such as sleeping, communicating (writing or video conferencing with remote family and friends) and entertaining (watching TV, listening to music etc).

The iDorm1 is fitted with many sensors and effectors to enable the both the inhabitants and the monitoring agent to observe and make changes to the room's environmental conditions. The sensor network includes devices such as: temperature sensors (both inside and outside the room), humidity sensors, a small matrix of light sensors across the room, an active entrance lock system which provides access based on an individual's identity, a infrared motion sensor, pressure sensors distributed in the room's furniture, etc. A number of actuators are also connected to the network and these include an air circulator, fan heater, door lock actuator, motorised vertical blinds, automated window openers, and a light dimmer.

The second testbed is called the iDorm2 (shown in figure 1). This environment is a newer development and is a much larger scale living space in the form of a two bedroom apartment with separate areas such as a kitchen, dining room, entertainment area and bathroom. Both environments have furniture based sensors but also sensors able to detect changes in temperature and humidity.

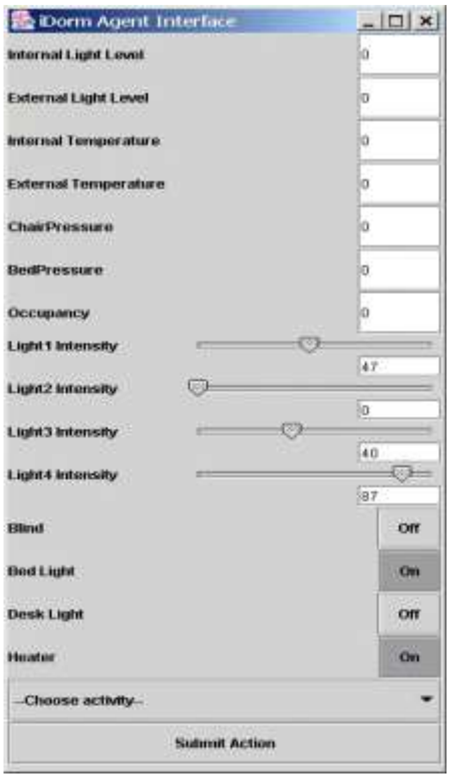
Figure 1. The iDorm-2

The sensors and actuators inside the environments need to communicate. For this purpose, Ethernet, overlaid with TCP/IP and UPnP middleware programmed in Java were used. The use of Java, presents some obvious advantages such as to be an object oriented language, providing greater flexibility, modularity and reusability. It was also chosen because of its portability and the possibility to be used in embedded devices such as the TINI and SNAP board (cheap embedded-internet devices). Being an interpretive language, Java will run slower than compiled languages; thereby through its use we also demonstrate that agents built with faster compiled languages, such as C, are also feasible.

The environment devices are connected to the agent using the UPnP (Universal Plug and Play) [20] middleware to communicate the actions inside the iDorm. To do this we "wrap" the sensors and actuators to provide them with an UPnP interface, connecting them to the control point using an IP network.

In the system presented here, the controlled devices correspond to the sensor and actuator devices (i.e. bed lamp, chair sensor, temperature sensor) and the control point role is performed by a PC running the UPnP stack and providing a user interface (UI) which is used to control the devices and to send the information about the state of the sensors and the actions performed by the user to the agent.

The user interface used to control the iDorm environment and to record the user's actions and behaviours is shown in figure 2. The UI displays information about the state of the different devices inside the iDorm and at the same time provides a way to modify the state of the environment and to record the activity being performed by the inhabitant at that moment.



| | |
|---|---|
| Internal Light Level | 0 |
| External Light Level | 0 |
| Internal Temperature | 0 |
| External Temperature | 0 |
| ChairPressure | 0 |
| BedPressure | 0 |
| Occupancy | 0 |
| Light1 Intensity | 47 |
| Light2 Intensity | 0 |
| Light3 Intensity | 40 |
| Light4 Intensity | 87 |
| Blind | Off |
| Bed Light | On |
| Desk Light | Off |
| Heater | On |

Figure 2 .User Interface

## 3.2. Agent hardware description

Two different agents, that use the information provided by the sensorial network, were programmed in Java. The first one was programmed on a PC and the second, on a SNAP board. The SNAP board is a network-ready, Java-powered plug & play reference platform designed for use in embedded computing applications [19]. The SNAP device has a Cjip microprocessor developed by Imsys which has been designed for networked, Java-based control. It runs at 80 MHz and has 8 Mb DRAM + 2 Mb RAM. It uses the J2ME (Java 2 Micro Edition) CLDC (Connected Limited Device Configuration) which is a very low-footprint Java runtime environment [21].

The main purpose of programming an agent to run on a SNAP board was to show that the agent was able to run on an embedded-internet device. As mentioned earlier, we used Java with its large computational overheads for the same reason. One of the goals of our activity and behaviour recognition system was that it should be small enough to fit on a real embedded device. The chosen network is able to recognize and detect behaviours inside an environment using very limited processor power.

In order to test the SNAP based agent with the iDorm data, a UDP connection was established between the UPnP control point and the SNAP based agent. The information was sent in real time and the agent was trained in an online way with each one of the incoming data instances.

## 3.3. Temporal network architecture

Two of the major concerns related to care environments are methods of recognizing and adverting medical crisis [13], and the lack of proper agent architectures (both the internal & external agent structures and mechanisms) able to cope with the challenges of more demanding scenarios.

Our system uses a neural-network based agent in order to detect, recognize and classify human activities and behaviours inside an environment  In order to use a neural network for this purpose, the network must be able to identify recurrent patterns of behaviour, yet flexible enough to adapt itself to continuous changes in the environment. Previous approaches have shown the feasibility of using neural network in intelligent environments [11] but they differ with the present work as this earlier work only used them for energy conservation management.

Our approach comprises the use of an Adaptive Neural Architecture derived from the ECoS paradigm proposed in [7]. This kind of network can grow dynamically, adapting its hidden layer to accommodate new information by adding nodes (rule nodes) whenever an example is not found to fit in the already existing structure. It is even able to "grow" new input or output nodes to accommodate, for example, new sensor inputs or new activities.

Many of the abnormalities in behaviours that can be detected are not only related with the appearance of new activities but also with the temporal order in which those take place. With the addition of memory structures, the learnt temporal associations can be used to support the activation of the rule nodes based on temporal patterns along with pre-existing spatial-similarity associations found in the activities and human behaviours.

An adaptation of an Elman [4] (a Recurrent Neural Network) architecture was chosen because recurrence lets the network remember information from the recent past and does not appreciably complicate the training. The addition of a temporal layer and the connection weights allows the network to capture temporal dependencies between consecutive data examples.

Figure 3 provides a diagram of the actual structure of the adaptive neural network, consisting of four layers: the input layer, the evolving hidden layer and the output layer, plus a memory layer used to represent temporal information. The dashed arrows represent fully connected layers whereas the solid line arrow represent one to one connection between neurons.

The memory layer is a structure that allows the network to have temporal capabilities [4]. The network has feedback connections from the hidden layer of neurons back to the same nodes, and these context units simply hold a copy of the activations of the hidden units from the previous time step. These context units or memory layer allow the network to capture temporal dependencies between the presented data examples from the data stream.

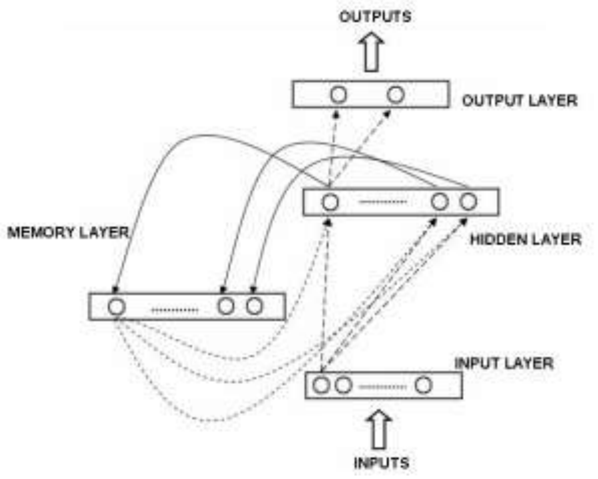


Figure 3. Neural network structure

The input layer is non-processing, so it only has linear activation functions. For each node in the hidden layer a normalised distance function (a Manhattan distance), is used. The distance function *D*, shown in equation 1 calculates the distance between the current input vector *I* and the incoming connection weight vector *Wih* (weights between the input and hidden layer) of a particular *hnode* (hidden node).

$$D_{I,hnode} = \frac{\sum_{k=1}^{n} \left| I_k - Wih_{k,hnode} \right|}{\sum_{k=1}^{n} \left| I_k \right| + \sum_{k=1}^{n} \left| Wih_{k,hnode} \right|} \quad where\ n = \#\,inputs \tag{1}$$

The activation function *A* of each neuron in the hidden layer is now dependant of both the spatial and temporal components. The proportion in which these two components influence the neuron activation can be modified by the spatial and temporal factors (*Sf* and *Tf* respectively). The activation function is calculated as shown in Equation (2):

$$A_{hnode} = 1 - \left( Sf * D_{I,hnode} + Tf * Wmh_{\max actv,hnode} \right) \tag{2}$$

Where $D_{I,hnode}$ represents the distance function between the input and a hidden node whereas $Wmh_{maxactv,hnode}$ represents the memory-to-hidden layer connection weight between the maximum activation neuron and the hidden node.

Only the winning, or most highly activated hidden neuron, is allowed to activate and the activation of the output neurons O*p*, shown in equation 3, is simply a product operation over the winning hidden neuron activation *A* and the hidden-to-output layer connection weights *Who*.

$$O_p = A_{hnode} \times Who_{hnode,p} \qquad (3)$$

The learning algorithm consists of placing the input examples within the hidden (evolving layer) by either creating a new node or modifying the connection weights. The algorithm is shown in figure 4.

```
For each input example presented to the network
    Calculate the error between the calculated outputs and the desired outputs
If the activation of the winning node is less than the Sensitivity threshold or the desired output node is not the most highly activated or the error is greater than the Error threshold
    Add a node
Else
    Update the connections of the evolving layer winning node
```

Figure 4. Adaptive algorithm

The input-to-hidden layer connection weights *Wih* are updated in an unsupervised way using equation 4 where $I_i$ is the input vector and $\eta_1$ is the learning rate.

$$Wih(t+1) = Wih(t) + \eta_1\ (I_i - Wih(t)\ ) \qquad (4)$$

The hidden-to-output layer connection weights *Who* are updated in a supervised way using equation 5, where $A_h$ is the activation of the winning hidden node, *Err* is the error between the desired and the calculated output and $\eta_2$ is the learning rate.

$$Who(t+1) = Who(t) + \eta_2(A_h \times Err) \qquad (5)$$

The memory-to-hidden layer connection weights *Wmh* capture the temporal dependencies between consecutive data examples.

These *Wmh* weights are updated using Equation (6) where $A_h$ is the activation of the winning hidden node, $A_m$ is the activation of the winning memory node and $\eta_3$ is the learning rate.

$$Wmh(t+1) = Wmh(t) + \eta_3\ (A_h \cdot A_m\ ) \qquad (6)$$

With the addition of the new memory structures, the learned temporal associations can be used to support the activation of the rule nodes based on temporal patterns along with the already existing spatial-similarity associations found in the activities and human behaviours.

## 4. Experiments and results

The experiments were divided into 2 major sections. The first one is related to the recognition of human activities or behaviours in order to assess the performance of the network for classifying activities and to compare it with other methods. The second one is oriented to test the ability of the network to spot novelties in those behaviours.

Due to the intrinsic characteristics of the smart living environments, such as the possibility to adapt themselves to their occupants and the presence of multiple sensors able to collect data from the users, the protection of users' personal information becomes a key issue [ ].

The user must be aware that data is being collected and stored, but only data relevant to the application is being recorded and used only for the purposes specified by the study.

For the experimental data needed for our experiments, the users were fully aware that data about their activities was being recorded and whenever they changed the activity being performed, they needed to capture that using a User Interface (UI). Moreover, the type of sensors (pressure, motion, light, etc) used in these experiments are regarded as non-intrusive to the user's privacy as no video or audio recording devices were used.

Two different datasets formed by the data collected by 18 sensors (both environmental and furniture based) were used. A set of eight different activities ('Listening to music', 'Working at Computer', 'Reading, Desk Work', 'Resting-napping, Sleeping', 'Out of Room', and 'Other') were detected inside the environment. In order to know which activity was being performed at a certain point of time, the user was asked to describe the action he/she was performing via a simple user interface, using an approach similar to the Experience Sampling Method (ESM) used in [12].

In order for the agent to collect the data regarding the user's behaviour and activities inside the environment, the agent recorded a 'snapshot' of the current inputs (sensor states) and the activity being performed at the time. This information constituted the input vector for the network and it has the following structure:

$$SS_1, SS_2, \ldots SS_n, Act \qquad (7)$$

Where *SS* denotes the state of each one of the sensors and *Act* the activity codified in an 1-of-N way. The information from the sensors can be continuous or binary. This input vector is then fed to the agent learning mechanism using a UDP connection or simply from a text file.

The 'snapshots' from the environmental state were acquired in two different ways. In the first one, a snapshot was taken when the user explicitly changed the state of the environment (e.g. turn on/off a switch), however, by using this approach, the number of instances recorded was limited (about 100 a day). By using this approach one dataset of 471 instances was collected over a period of 4 days. This dataset was known (and referred to in the following sections) as 'dataset A'.

The second dataset was formed by recording the state of the environment every 30 seconds regardless of the user's interaction with the system. By using this approach, a far larger dataset could be collected. In this case, the user was asked to live inside the intelligent environment for a 6-day period and approximately 8,000 instances were recorded. This dataset was known (and referred to in the following sections) as 'dataset B'.

## 4.1. Activity recognition

**4.1.1. Comparison using different approaches.** The first step towards the detection of abnormal activities inside an environment is to detect and recognize a normal set of activities. The use of both unsupervised and supervised approaches was explored in the early stages of this research. A series of experiments using the 'dataset A' were conducted in order to compare the performance of different approaches in the activity recognition problem. The activities 'dataset A' was divided into 314 examples for training and another 157 examples were used for testing. The results reported below show the accuracy of different approaches when evaluated over the testing set.

| Method | Correct | Incorrect |
|---|---|---|
| EM Clustering | 38.85 % | 61.15 % |
| KMeans | 57.54 % | 42.46% |
| N.N. Competitive Learning | 47.77 % | 52.23% |
| Naïve Bayes | 76.00 % | 24.00 % |
| MultiLayer Perceptron (1) | 92.36 % | 7.64 % |
| MultiLayer Perceptron (2) | 47.98 % | 52.012 % |
| Adaptive Neural System | 91.08 % | 8.92 % |

**Table 1 Comparison between different methods in activity recognition**

The first three methods made use of unsupervised learning approaches. They were tested in order to investigate the viability of unsupervised learning for activity recognition as these methods do not rely on input from the inhabitants in order to separate the activities performed in the environments. These methods didn't show a very good performance, with recognition rates lower than 50%. These results, however, might be improved if the number of activities performed and the number of sensors used were increased, so the possibility of using some kind of unsupervised learning should not be excluded.

Our system was also compared with other off-line supervised techniques in order to analyse its performance (using tenfold cross validation over the 471 instances of 'dataset A'). First, a test using the Naïve Bayes method was performed so as to use it as a benchmark since the vast majority of the existing activity recognition tools use a Bayes-related method.

Two different tests were conducted using a MultiLayer Perceptron (MLP), the first by training the network for 500 epochs and the second one training it only for 1 epoch. Our system performed better than the Bayes classifier, clearly outperforming the MLP trained for 1 epoch and showing a similar performance than the MLP trained for 500 epochs. It should be noted that our system was trained simulating on-line learning that is, using one-pass learning in which the examples were presented to the network just once. Moreover, our system could be extended using fuzzy logic in order to deliver human readable rules so it is possible to look at the relationships between the sensors and the activities.

**4.2.2. Further activity recognition tests.** In order to fully test and validate the results of the agent, more experiments were performed using the data from 'dataset B'. The agent was tested using the data acquired during the first 4 days of experiments. The first tests were done using fourfold cross-validation where the 4 partitions corresponded to the data of each one of the 4 days; the network was tested 4 times being trained by 3 days of data and tested using the remaining day. The total number of instances for the 4 days was approximately 5,500. The results are summarized in the table below:

|  | Train | Test |
|---|---|---|
| 1st run | 97.8 % | 89.0 % |
| 2nd run | 98.0 % | 85.3 % |
| 3rd run | 97.5 % | 71.68 % |
| 4th run | 97.5 % | 77.83 % |
|  |  |  |
| Average | 97.95% | 80.95% |

**Table 2 Classification of activities**

As shown in the above table, for this set of experiments, the results were satisfactory, and similar to the ones obtained with other users. It can be seen that the test results for the last two series of the experiments were different to first two (80% vs. 70% of accuracy). In order to find if this difference was caused by different distributions of activities inside the datasets, the test was performed again, this time using a stratified fourfold cross-validation. The stratified cross-validation divides the whole dataset in 4 folds, each one with the same class distribution. The results are shown in the table below:

| | Train | Test |
|---|---|---|
| 1st run | 96.9 % | 98.9 % |
| 2nd run | 96.9 % | 98.6 % |
| 3rd run | 97.2 % | 98.6 % |
| 4th run | 97.0 % | 96.9 % |
| | | |
| Average | 97.0 % | 98.0 % |

**Table 3. Classification of activities II**

As it can be seen in table 3, much better results were obtained if a stratified cross-validation was used. It is very possible that the class distribution (the distribution of the different types of activities along a certain day) plays an important role in the correct classification and detection of patterns in human behaviour inside a ubiquitous environment.

## 4.2. Novelty detection

**4.2.1. Comparison tests.** Two experiments were performed in order to know if the system was able to detect novelties on datasets previously used as benchmark; the results are described in this section.

*The iris dataset*

The first experiment to detect novelties was conducted using the well-known iris data set. This dataset consists in three different classes, with four attributes, with 50 examples of each one of them. Two different tests following the same approach were completed. 40 randomly chosen examples per class from two classes were used as a training set (80 in total) and the remaining 10 examples per class of those 2 classes along with the 50 examples of the other class (the "abnormal" class) were used for testing the network for anomaly detection. The results are summarized in the following table:

| Testing | Abnormal class | Class 3 | Class 1 |
|---|---|---|---|
| | Detected novelties | 53 | 50 |
| | True novelties | 49 | 50 |
| | Mislabelled novelties | 4 | 0 |

**Table 4 Abnormality detection in iris dataset**

From the results above, it can be observed that in the first test 49 out of 50 anomalies were detected, but 4 novelties were wrongly labelled. In the second test 50 out of 50 anomalies were detected with no mislabelled examples.

In the iris dataset, classes 2 and 3 can be linearly separated from class 1, but they are not linearly separable between them, so that could be the reason why in the first test 4 examples were mislabelled. Nevertheless, both tests showed that the novelty detection method was very good at detecting examples of a class never seen before by the network, therefore, considered as novel or abnormal.

*The biomed dataset*

The second experiment was conducted using the biomed dataset available at [12]. This dataset [3] has been used as a benchmark in abnormality detection, and has been previously used by Campbell and Benett [2] and Marsland [8]. The dataset consists of 194 observations each with 4 attributes corresponding to measurements made on blood samples (15 observations with missing values were removed/data has been normalised). The normal data consists of 100 observations made on normal healthy patients. The test sets consist of 27 observations made on normal healthy patients and 67 exhibiting abnormalities due to a rare genetic disease.

This experiment was done in order to assess the performance of the network against other existing techniques and to test the two different methods of novelty detection (threshold in the hidden layer, and threshold in the distance between the calculated output and the entire training output pattern). The results are summarized in table 5:

| Testing | Testing examples. | 94 |
|---|---|---|
| | Detected novelties | 63 |
| | True novelties | 57 |
| | Mislabelled novelties | 6 |

**Table 5 Biomed dataset results**

The results showed that our ECoS system performed similarly to other methods. Marsland [8] reports 56 of the 67 novel inputs highlighted as novel, with 2 examples mislabelled, Campbell et al. [2] report 57 of the 67 novel inputs correctly labelled as novel with again 2 mislabelled examples. Our system found 57 out of 67 examples correctly labelled as novelties and showed up 6 mislabelled examples. As it has been discussed in [8] the main purpose of a novelty detector is to avoid false negatives (i.e. not detecting possible anomalies) while trying to highlight all the abnormal examples in the data. Although our system reported 6 false positives (i.e. examples incorrectly labelled as abnormal) that is not a major issue because for novelty detection systems it is better to detect as many abnormalities as possible even if some examples are incorrectly labelled as abnormal (providing that those false positives are not too many).

**4.2.2. Novelty detection with activities dataset.** This experiment was performed using the activities 'dataset A'. Two different trials were conducted, both of them using a threshold in the hidden layer to find the novelties. The training and test sets were built in the following way: First, one class was removed from the dataset, and 300 of the remaining randomised examples were used as the training set. The removed class and the remaining examples constituted the test set.

| | | | |
|---|---|---|---|
| Testing | Training examples | 171 | 171 |
| | Abnormal class | Out of room | Desk work |
| | Number of abnormal examples | 83 | 55 |
| | Detected abnormalities | 84 | 56 |
| | True abnormalities | 81 | 42 |
| | Mislabelled | 3 | 14 |

**Table 6 Activities dataset**

As can be observed in Table 6, the network finds 97.6% of the "Out of room" examples to be abnormal, wrongly labelling as abnormal 3.41% of the normal data. In the second experiment 76.3% of the "Desk work" examples, were correctly labelled as abnormal and 12.0% of the normal data was mislabelled.

The reason why the results were better for the "Out or room" class can be traced back to the results of the network in the activity recognition experiment. That class was one of the best classified activities while the "Desk work" was the second worst classified. As it was observed in Table 2 and 3 the class distribution (the distribution of the different types of activities along a certain day) plays an important role in the correct classification of activities and it also affects the novelty detection process.

**4.2.3. Temporal abnormality detection.** Two different experiments were performed in order to test the ability of the network to spot abnormalities related with time. In the first experiment, the network was tested to detect an activity that has been previously seen but that this time has presented itself at a different hour of the day. A graphical representation of the normal order of the activities (Figure 5) and one with the activity taken place at a different hour (Figure 6) are both shown in the figures below with the hour in the Y-axis an the activities on the X-axis.

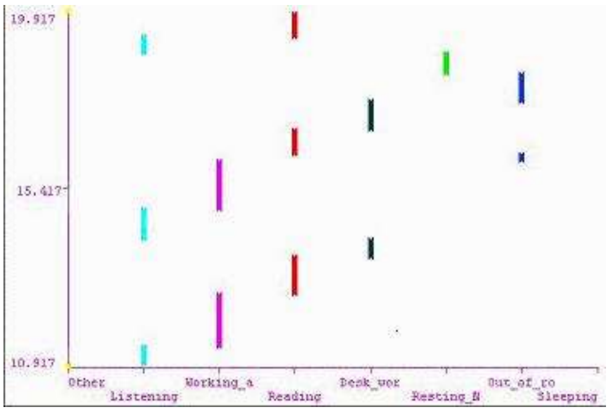


Figure 5. Normal activities

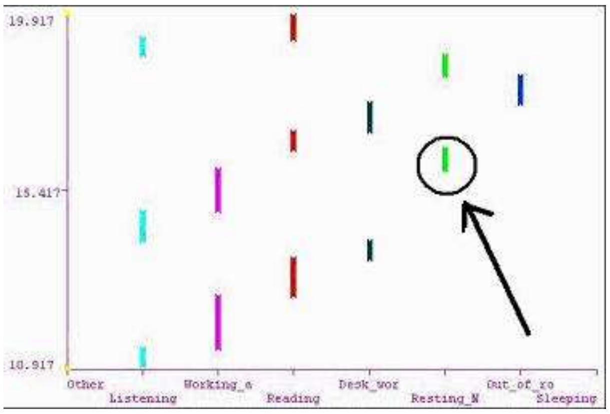


Figure 6. Activity taking place at a different time

The network was tested in order to see if it can spot an activity taking place at an unusual time according to the normal pattern of behaviour. This abnormality consisted on the inhabitant taking a nap at a different hour. The results are summarized in the table below:

| | | |
|---|---|---|
| **Testing** | Novelty sensitivity thr. | 0.95 |
| | Detected novelties | 31 |
| | True novelties | 31 |
| | Mislabelled novelties | 0 |

**Table 6 Abnormality detection for activities occurring at different time**

The network found 31 abnormal activities corresponding to the 31 instances in which the napping activity occurred at a different time, correctly spotting 100% of the abnormal instances. More experiments need to be conducted in order to get more conclusive results however; the network seems to be well suited for this kind of task.

For the second experiment the network was tested to spot abnormalities on activities occurring with a different order. The comparison again was made using a normal set of activities shown in Figure 5 and the new order of the activities can be seen better in the graphical representation presented in Figure 7.

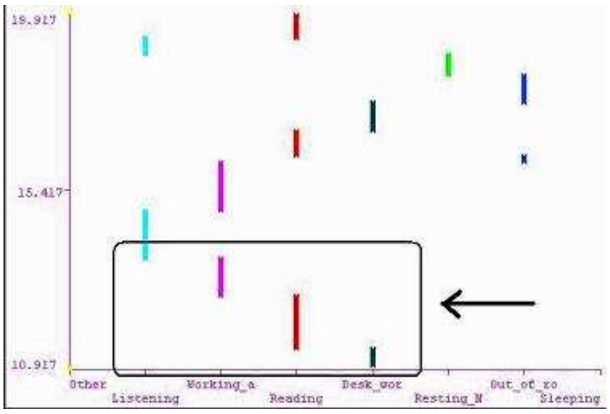


Figure 7. Activity taking place at a different time

| | | |
|---|---|---|
| **Testing** | Novelty sensitivity thr. | 0.98 |
| | Total novelties | 182 |
| | Detected novelties | 184 |
| | True novelties | 122 |
| | Mislabelled novelties | 62 |

**Table 7 Abnormality detection for activities occurring at different time**

The network finds 122 true abnormalities, corresponding to the 67% of the total of the real abnormalities, however almost all the abnormalities that were not found corresponded to the "Working at computer" This class shared similar sensorial activations with other activities, thus its higher failure ratio.

The results of the two experiments have shown that the temporal network is able to find abnormalities in which time plays a significant role.

## 5. Future work and Conclusions

The system presents a novel system which employs an innovative neural network running in an embedded agent. This agent works with online data and is able to classify activities and spot abnormal patterns of behaviour in real time. Tests have been conducted both in simulated and in real environments, showing that the system can cope with the temporal aspect of human behaviour in an automated way.

The results show that the network architecture can achieve very good recognition rates, better than the ones obtained while using Bayesian approaches (which have been often used for the activity recognition problem) and comparable to the ones obtained with other neural networks but with the advantage that our approach can be trained online and continuously and can be fitted into very small processors, using limited hardware resources.

To our knowledge, there aren't any agents, capable of recognizing human behaviours inside an environment, that have been implemented within a small, cheap, off-the-shelf embedded processor of the type that could be readily commercially deployed. The development of AI techniques that can be run inside in such a device is by itself an important contribution of this work.

The novelty detection ability of the network has been compared with other approaches in the literature and has shown very similar results. It also has proven to be well-suited for novelty detection in human behaviours.

Concerning our future plans, as a result of the encouraging results reported in this paper, we are planning a larger scale experiment with many more sensors, actuators and users so as to explore more finely grained behaviour sensing using both a simulated environment and a real environment.

## Acknowledgements

F. Rivera-Illingworth wishes to thank the CONACYT (the Mexican National Council of Science and Technology) for its funding support.

## References

[1] Abowd, G., Mynatt, E., Rodden, T. (2002) The Human Experience - Reaching for Weiser's Vision. IEEE Pervasive Computing, 1(1), 48-57.
[2] Beaudin, J., Intille, S., and Munguia, E. (2004) Lessons learned using ubiquitous sensors for data collection in real homes. Proceedings of CHI '04 extended abstracts on Human factors in computing systems, Vienna, Austria,24-29 April, pp. 1359-1362. ACM Press, New York, NY.
[3] Chappelier, J., Gori, M., and Grumbach, A. (2001) Time in Connectionist Models, Lecture Notes in Computer Science, 1828, 105-134.
[4] Elman, J. (1990) Finding structure in time. Cognitive Science, 14(2), 179-211.
[5] Hertz, J., Krogh, A., and Palmer, R.G. (1991) Introduction to the Theory of Neural Computation, Addison-Wesley Publishing Company, California.
[6] Jimison, H., Pavel, M., and Pavel, J. (2003) Adaptive Interfaces for Home Health. Proceedings of UbiHealth 2003: The 2nd Int. Workshop on Ubiquitous. Computing for Pervasive Healthcare Applications, Seattle, WA, 12 October. CD-ROM.
[7] Kasabov, N. (2002) Evolving connectionist systems: Methods and applications in bioinformatics, brain study and intelligent machines, Springer, London.
[8] Kremer, S. C. (2001) Spatio-temporal Connectionist Networks: A Taxonomy and Review. Neural Computation, 13(2), 249-306.
[9] Mihailidis, A., Carmichael, B., Boger, J., and Fernie, G. (2003) An Intelligent Environment to Support Aging-in-Place, Safety, and Independence of Older Adults with Dementia. Proceedings of UbiHealth 2003: The 2nd Int. Workshop on Ubiquitous. Computing for Pervasive Healthcare Applications, Seattle, WA, 12 October. CD-ROM.
[10] Mozer, M. (1994) Neural net architectures for temporal sequences processing. In Weigend, A. and Gershenfeld, N.(eds), Time Series Prediction: Forecasting the Future and Understanding the Past. Addison-Wesley, Reading, MA.

[11] Mozer, M. (1998) The Neural Network House: An Environment that Adapts to its Inhabitants. Proceedings of the AAAI Spring Symposium on Intelligent Environments, pp. 110-114. AAAI Press, Menlo Park, CA.
[12] Munguia, E., Intille, S., and Larson, K. (2004) Activity Recognition in the Home Using Simple and Ubiquitous Sensors. Proceedings of PERVASIVE 2004, Vienna, Austria, 21-23 April, pp. 158-175. Springer-Verlag, Berlin.
[13] Mynatt E., and Rogers, W. (2002) Developing technology to support the functional independence of older adults. Ageing International, 27(1), 24-41.
[14] Philipose, M., Fishkin, K., Perkowitz, M., Patterson, D., Fox, D., Kautz, H., and Hähnel, D. (2004) Inferring Activities from Interactions with Objects. IEEE Pervasive Computing, 3(4), 50-57.
[15] Rivera-Illingworth, F., Callaghan, V., Hagras, H. (2004) A connectionist embedded agent approach for abnormal behaviour detection in intelligent health care environments. Proceedings of Systems, Man and Cybernetics, 2004 IEEE International Conference, The Hague, Netherlands, 10-13 October, pp. 3565-3570, IEEE.
[16] Sixsmith, A., and Johnson, N. (2004) Smart sensor to detect falls of the elderly. IEEE Pervasive Computing, 3(2), 42-47.
[17] Wilson, D., and Atkeson, C. (2005) Simultaneous Tracking and Activity Recognition (STAR) Using Many Anonymous, Binary Sensors. Proceedings of PERVASIVE 2005, Munich, Germany, May, pp. 62-79. Springer-Verlag, Berlin.
[18] Haigh, K., and Yanco, H. (2002) Automation as Caregiver: A Survey of Issues and Technologies. Proceedings of AAAI-02 Workshop on Automation as Caregiver: The Role of Intelligent Technology in Elder Care, Edmonton, Alberta, 29 July pp. 39-53. AAAI Press.
[19] http://www.imsys.se/
[20] http://www.plug-n-play-technologies.com/index.htm
[21] http://java.sun.com/products/cldc/
[22] http://familydoctor.org/662.xml Dementia : What are the common signs? – familydoctor.org
[23] http://www.alz.org/AboutAD/10Signs.htm About Alzheimer's | Ten Warning Signs of Alzheimer's disease.
Kidd, C., Orr, R., Abowd, G., Atkeson, C., Essa, I., B. MacIntyre, B., Mynatt, E., Starner, T., and Newstetter, W. (1999) The Aware Home: A Living Laboratory for Ubiquitous Computing Research. Proceedings of the Second International Workshop on Cooperative Buildings. 01-02 October, pp. 191-198. Springer-Verlag, London.

Smart Medical Home Research Laboratory at University of Rochester, http://www.futurehealth.rochester.edu/smart_home/

Cook, D. and Das, S. (2007) How smart are our environments? An updated look at the state of the art. Pervasive and Mobile Computing, 3(2) , 53-73

Porteus J., Brownsell S. (2000) Using Telecare: Exploring Technologies for Independent Living for Older People. Anchor Trust, Kidlington.

Stanford, V. (2002) Using Pervasive Computing to Deliver Elder Care. IEEE Pervasive Computing, 1(1), 10-13.

Nixon, P. Wagealla, W., English, C., and Terzis, S. (2004) Privacy, Security, and Trust Issues in Smart Environments. In Cook, D. and Das, S. (eds), Smart Environments: Technology, Protocols and Applications. Wiley, London.

Callaghan V, Clarke G, Chin J "Some Socio-Technical Aspects Of Intelligent Buildings and Pervasive Computing Research, Intelligent Buildings International Journal, Vol 1 Issue 1, 2007