Development of Cognitive Capabilities for Smart Home using a Self-Organizing Fuzzy Neural Network

Anjan Kumar Ray, Gang Leng, T. M. McGinnity, Sonya Coleman, Liam Maguire

Intelligent Systems Research Centre (ISRC), Faculty of Computing and Engineering, University of Ulster, Magee Campus, Londonderry, BT48 7JL, U.K. (e-mail: {ak.ray, g.leng, tm.mcginnity, sa.coleman, lp.maguire}@ulster.ac.uk)

Abstract: A smart home requires cognitive assistance to analyze and understand the behavior in this sensory rich environment. In this paper we explore the potential of a self-organizing fuzzy neural network (SOFNN) as a core component of a cognitive system for a smart home environment. We develop a cognitive reasoning module that has the ability to adapt its neuronal structure through adding and pruning of neurons according to the incoming data. The SOFNN rules explore the relations of the inputs and the desired reasoning outputs. The network is trained with realistic synthesized data to show its adaptation capability and is tested with unseen data to validate its cognitive capabilities. We outline the theoretical development and describe the results achieved. This initial implementation of the cognitive module demonstrates the potential of the architecture and will serve as a very important test-bed for future work.

Keywords: Artificial intelligence, self-organizing system, fuzzy system, neural network, cognitive system

1. INTRODUCTION

A smart home is a sensor rich environment that provides several technical innovations to improve the daily life of its residents. To develop a smart home researchers are exploring solutions for low-level data collection, information processing and high level service delivery. Gaddam et al. (2011) report a home monitoring system based on a cognitive sensor network for elderly-care application. Processing of this sensory information is necessary to understand the context of the ecology. Wang et al. (2005) proposed a context-aware system, CASSHA (Context-Aware System for Smart Home Applications), for processing, representation, provision, and coordination of smart home applications. Mastrogiovanni et al. (2010) have integrated ontology and logic based approaches to context representation and recognition to map numerical data to symbolic representations. A resource-aware smart home management system is reported by Son et al. (2011). The intelligence of the smart home comes from the adaptive behaviour of the overall ecology as per the requirement of the user. Roy et al. (2010) have reported an initial framework of activity recognition based on possibility theory and description logic (DL) as the semantic model of the agent's behaviour. Chen (2010) has discussed the concept of semantically enhanced situation awareness and intelligent just-in-time Activity of Daily Living (ADL) assistance provision within integrated system architecture. The situation awareness within a smart home also needs to detect anomalous events. Jakkula (2011) has used One Class Support Vector Machines (OCSVM) techniques to address this issue and reported an initial set of results. The main objectives of introducing intelligence into a smart home environment are to identify events that require a response, to understand the extent of the event's importance and to propose or automatically activate a suitable response (Bregman, 2010). To address these issues, we have proposed a cognitive reasoning module that analyzes the events of a smart home ecology and reasons across those events to build the situational awareness. This initial work is the first step towards the autonomous adaptation of the ecology. A Self-Organising Fuzzy Neural Network (SOFNN) based machine learning technique, developed by the authors (Leng et al. (2005)), is proposed for this work. The main advantages of the SOFNN are automatic identification of the structure and parameters of fuzzy neural networks from data, and evolution and update of the structure and parameters by using new data to follow the dynamic changes in the environment. Leng et al. (2012) have reported that SOFNN performs better than ANFIS (Jang (1993)), RBFAFS (Cho (1996)), GDFNN (Wu et al. (2001)), and FAOS-PENN (Wang et al. (2009)).

The remainder of the paper is organised as follows: section 2 describes the details of the design and implementation issues of the cognitive reasoning module along with neuron adding and pruning strategies. A sliding window control method is also implemented to examine the effect of historical data on the current reasoning output. Section 3 presents the implementation results of the proposed work. A set of anticipated events and reasoning outputs are chosen to perform the validation. The results on structural growth of the SOFNN and the cognitive reasoning capabilities under a synthesized scenario are presented. Section 4 presents the overall conclusion of this work.

2. PROPOSED COGNITIVE REASONING MODULE

Figure 1 shows a general view of a smart home that hosts different sensors, actuators and robots. These sensors and actuators provide a set of on-going events within this environment. The proposed work is part of the RUBICON (Robotic UBIquitous COgnitive Network) EU FP7 project (RUBICON, 2011) which aims at creating a self-sustaining, self-organizing, learning and goal-oriented robotic ecology that consists of a network of heterogeneous computational nodes interfaced with sensors, effectors and mobile robots. There are four technical layers for this ecology named learning, control, communication and cognitive layers. The learning layer addresses sensory information for event classification, the control layer employs robots for different goals within the ecology whereas communication layer is responsible for data transmission among the layers. The cognitive layer focuses on the knowledge development that accurately reflects the dynamics of the ecology and cognitive capability. To demonstrate the cognitive capability, it is necessary to handle multiple events that may occur in the ecology, and in particular extract higher-level intelligence from apparently conflicting, overlapping or supporting events in the ecology. In this work, we report the development of these cognitive capabilities through a cognitive reasoning module as shown in Fig. 2. The processed sensory/event information is the input for this structure that comes with a confidence level between 0 and 1 portraying non-occurrence or occurrence of that event. The reasoning module is developed as a multi-input multi-output (MIMO) SOFNN structure as shown in Fig. 3 where each sub-module is responsible for one desired reasoning output. The desired reasoning outputs are available in the reasoning goals table. A window control is also employed to verify the effect of previous event data on the current reasoning outputs. The 5-layer structure of an individual SOFNN module is shown in Fig. 4 where x_1, x_2 ,… x_r are input variables and y is the reasoning output.

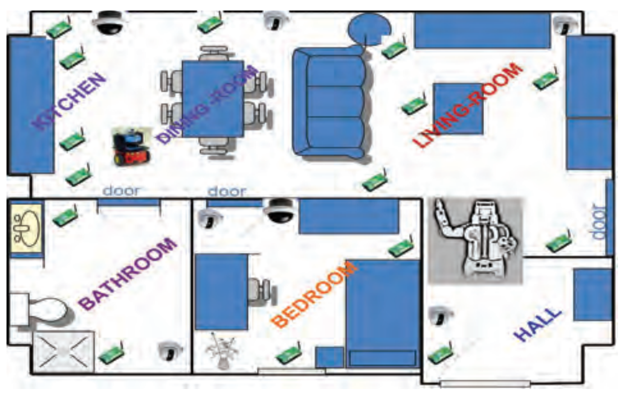

Fig. 1. An outline of a smart home with sensors and robots

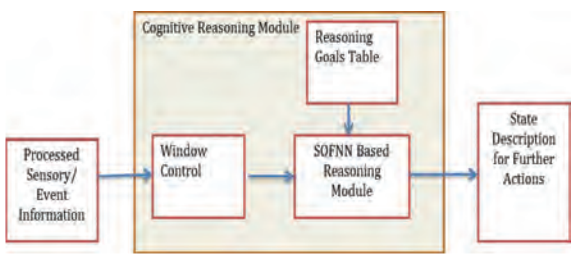

Fig. 2. Focused area of cognitive development

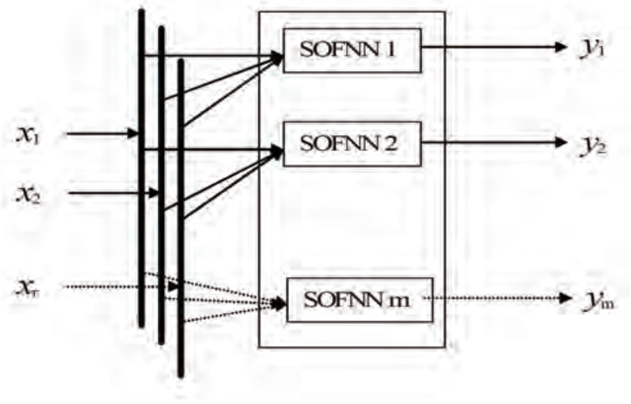

Fig. 3. MIMO structure of the first version of the cognitive reasoning module

2.1 General Structure of SOFNN

In this paper, the proposed network implements a first order Takagi-Sugeno (TS) model (Takagi, 1985) as a set of fuzzy rules of the following form:

R_i: If x_1 is A_{i1} and x_2 is A_{i2} and ⋯ and x_r is A_{ir}, then

$$y_i = a_{i0} + a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{ir}x_r \, , i = 1,2,..,u \tag{1}$$

where R_i represents the i-th rule, A_{i1}, A_{i2} , … A_{ir} are fuzzy sets, $y_i = a_{i0} + a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{ir}x_r$ is a linear crisp function of input variables, u is the number of fuzzy rules and r is the number of input variables.

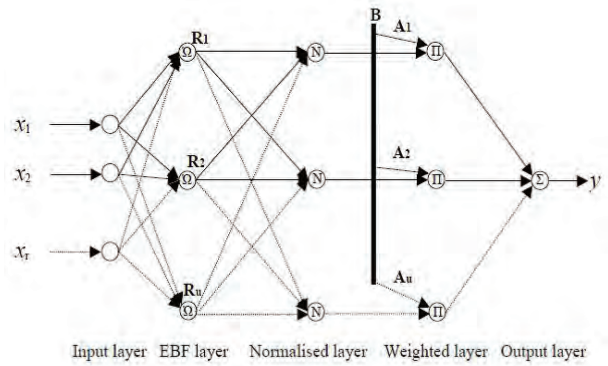

Fig. 4. Structure of individual SOFNN module

In Fig. 4, we can see that layer 1 is the input layer. Layer 2 is the ellipsoidal basis function (EBF) layer where each neuron represents an if-part (or premise) of a fuzzy rule. The output of each EBF neuron is the firing strength of each rule, and is given by

$$\phi_j = \prod_{i=1}^{r} \mu_{ij} = \prod_{i=1}^{r} \exp\left[-\frac{(x_i - c_{ij})^2}{2\sigma_{ij}^2} \right] \tag{2}$$

where i= 1, 2, … r; j= 1, 2, … u; μ_{ij} is the i-th membership function in the j-th neuron; c_{ij} and σ_{ij} are the centre and width of the i-th membership function in the j-th neuron respectively. The output from the layer 2 is normalized in Layer 3. The output of each neuron in this layer is given by

$$\psi_j = \frac{\phi_j}{\sum_{k=1}^{u} \phi_k} \tag{3}$$

Layer 4 is the weighted layer where the weighted bias is given by

$$w_{2j} = A_j B = a_{j0} + a_{j1}x_1 + \ldots + a_{jr}x_r \tag{4}$$

The bias for the TS model is given by $B = [1\ x_1\ x_2\ \ldots\ x_r]^T$ and $A_j = [a_{j0}\ a_{j1}\ \ldots\ a_{jr}]$ represents the set of parameters corresponding to the then-part (or consequent) of the *j*-th fuzzy rule. The output of each neuron is obtained as

$$f_j = w_{2j}\,\psi_j\,. \tag{5}$$

Layers 2, 3 and 4 consist of equal number of neurons. The contributions of the outputs from layer 4 are summed for the final output from the network within layer 5 as

$$y(x) = \sum_{j=1}^{u} f_j \tag{6}$$

The implementation of a SOFNN can be divided into parameter matrix updating, neuron adding and pruning strategies.

2.2 Parameter Matrix Learning

For training purposes, the output of the network can be described in matrix form for n training patterns of the input vector $x_t=[x_{1t}\ x_{2t}\ \ldots\ x_{rt}]$ and the corresponding desired output d_t ($t = 1,2,\cdots\cdots,n$) as

$$Y = W_2 \Psi \tag{7}$$

where W_2 is the parameter matrix represented as $W_2 = \left[a_{10}\ \ a_{11} \cdots a_{1r} \cdots a_{u0}\ \ a_{u1} \cdots a_{ur}\right]$, Ψ_{jt} is the output of the *j*-th neuron in the normalised layer associated with the *t*-th input vector, $Y = [y_1\ y_2\ \cdots\ y_n]$, and

$$\Psi = \begin{bmatrix} \psi_{11} & \cdots & \psi_{1n} \\ \psi_{11}x11 & \cdots & \psi_{1n}x1n \\ \vdots & \vdots & \vdots \\ \psi_{11}xr1 & \cdots & \psi_{1n}xrn \\ \vdots & \vdots & \vdots \\ \psi_{u1} & \cdots & \psi_{un} \\ \psi_{u1}x11 & \cdots & \psi_{un}x1n \\ \vdots & \vdots & \vdots \\ \psi_{u1}xr1 & \cdots & \psi_{un}xrn \end{bmatrix} \tag{8}$$

The parameter matrix W_2 is derived according to the following recursive parameter matrix learning algorithm (Leng, 2005),

$$L(t) = Q(t)p(t) = Q(t-1)p(t)\left[1 + p^T(t)Q(t-1)p(t)\right]^{-1} \tag{9}$$

$$Q(t) = \left[I - \alpha L(t)p^T(t)\right]Q(t-1) \tag{10}$$

$$\hat{\Theta}(t) = \hat{\Theta}(t-1) + \alpha L(t)[d_t - p^T(t)\hat{\Theta}(t-1)] \tag{11}$$

$$\alpha = \begin{cases} 1, & |e(t)| \geq |\varepsilon(t)| \\ 0, & |e(t)| < |\varepsilon(t)| \end{cases} \tag{12}$$

where $Q(t) = \left[P^T(t)P(t)\right]^{-1}$ is an $M \times M$ Hermitian matrix (Q-matrix), $P(t) = \Psi^T = \left[p^T(1)\,p^T(2)\cdots p^T(t)\right]^T$, $\Theta(t) = W_2^T = \left[\theta_1\ \theta_2\ \cdots\theta_M\right]^T$, $e(t) = d_t - p^T(t)\hat{\Theta}(t-1)$ is the estimation error, $M = u \times (r+1)$, and $\varepsilon(t) = d_t - y_t = d_t - p^T(t)\hat{\Theta}(t)$ is the approximation error.

2.3 Addition of Neurons in the Structure

At the beginning, the first EBF neuron is created based on the first input vector. The number of membership functions generated by each EBF neuron is the same as the number of inputs. To add a new neuron, two criteria, system error criterion (13) and if-part criterion (14), must be satisfied simultaneously.

$$|d_n - y_n > \delta| \tag{13}$$

$$\phi(n) = \max(\phi_j) > \varepsilon \tag{14}$$

where δ is the error tolerance and ε is the threshold for the output of each EBF neurons. Considering the input x_i at $c_{ij} \pm 2\sigma_{ij}$, from (2) we get ε is equal to 0.1354 which ensures that 95% of input data belonging to this membership function will fall within the input range $\left[c_{ij} - 2\sigma_{ij}, c_{ij} + 2\sigma_{ij}\right]$. Fig. 5 shows the algorithm for adding new EBF neurons to the existing structure.

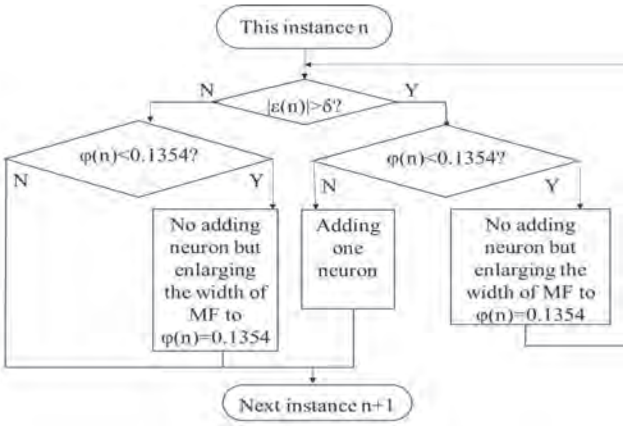

Fig. 5. The process of adding new EBF neuron

Membership functions in the k-th new EBF neuron can be defined as

$$c_k = [c_{1k}\ c_{2k}\ \cdots c_{rk}]^T \quad (15)$$

$$\sigma_k = [\sigma_{1k}\ \sigma_{2k}\ \cdots \sigma_{rk}]^T \quad (16)$$

where c_{ik} is the current input x_i, and σ_{ik} is the distance between the input x_i and the centre of the nearest membership function.

2.4 Pruning of Neurons

Neurons which do not affect the network's performance are deleted for model compactness. The order of neurons in terms of importance is decided based on the value of ΔE given by

$$\Delta \mathrm{E}(\Theta, t) \approx \frac{1}{2} \Delta\Theta^T H \Delta\Theta \quad (17)$$

where $H = Q^{-1}(t)$ is the Hessian matrix. The objective is to delete the least important neuron, then check the performance of the new network. If the training root mean squared error (RMSE) is less than a predefined value E, which depends on the specific problem and experience, then this neuron should be deleted. Then, the second least important neuron is checked, and so on. If the training RMSE is not less than this predefined value, then this neuron is kept and no further neurons are deleted. The flowchart for the pruning strategy is shown in Fig. 6.

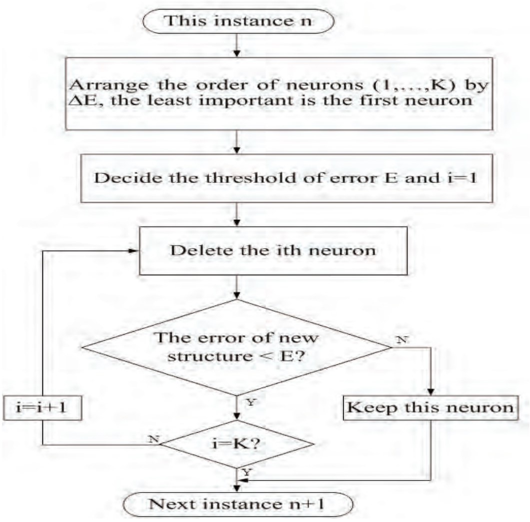

Fig. 6. The process of pruning neurons

2.5 Sliding Window Control

The sliding-window (SW) has been employed for online training and presents good performance (Ferreira, 2009), (Izzeldin, 2011). In this proposed work, the size of the Hermitian matrix (Q-matrix) depends on the number of neurons. If there is an addition or pruning of neuron, then the Q-matrix has to be updated based on limited historical and current data. The proposed window control strategy implements a first-in-first-out sliding window (FIFO-SW) and is shown in Fig. 7.

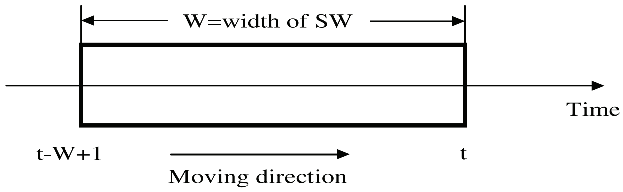

Fig. 7. Window control with FIFO-SW

A new structure is generated as per the addition and pruning of neurons. The parameters of the new structure are updated by

$$\hat{\Theta}(t) = \left[P^T(t)P(t)\right]^{-1} P^T(t)D(t) \quad (18)$$

where $D(n) = [d(1)\, d(2) \cdots d(n)]^T$. After that, the recursive parameter matrix learning algorithm is applied to update the parameters.

3. RESULTS

3.1 Anticipated Inputs

Typically, the cognitive layer will receive event data from the learning layer at each sample time, however at this stage; we synthesize the event data while considering variability of its confidence level ($\in$[0, 1]). For example, occurrence of a cooking event may not always represent a constant confidence level but assumes a range of confidence values over a period of time. We anticipate that some events occur individually and are independent to other events (e.g. the user may receive phone call which is not dependent on user's cooking activity) and some events are interrelated (e.g. cooking and dishwashing). The event data is generated to ensure a richness of variability with sufficient complexity to exercise the reasoning capabilities of the SOFNN system. We have anticipated 19 events as inputs from a home environment reflecting activities of a user and the states of the environment to test the cognitive capabilities. Table 1 shows the chosen inputs. In the event of multiple users in the home environment, the number of user related inputs and targeted outputs will increase.

3.2 Cognitive Reasoning Outputs

To enable the training process requires a set of desired reasoning outputs associated with the events. These outputs are generated by mimicking a human interpretation of the received information. For example, a user generally exercises and listens to music at the same time. As a result a set of nominal rules are created for the SOFNN which seed the training set with outputs associated with input patterns. These rules are generic in nature to provide a starting point for the SOFNN to explore and develop its reasoning capability. The output data are thus provided as confidence values which are time stamped for each possible output for a given set of input data. A set of 10 outputs is chosen to reflect the network's capabilities of reasoning across user activities, current state of the ecology and refinement of events as perceived by the learning layer; these are listed in Table 2.

Table 1: Event table: the inputs for reasoning module

Synthesized Input	Events
1	User in room 1
2	User in room 2
3	User in room 3
4	Visitor detection
5	Phone event
6	Doorbell event
7	Dripping event
8	Music event
9	Fire alarm
10	Microwave usage
11	Dishwasher usage
12	TV usage
13	Cleaning operation
14	Cooking
15	Use of oven
16	Smoke detection
17	Room temperature
18	Burglary alarm
19	Front door usage

Table 2: Targeted output of SOFNN reasoning

ID	Potential reasoning outputs
1	User exercise
2	User relaxing
3	User in kitchen
4	Bring phone situation
5	Open door situation
6	User's cooking activity
7	Fire alert situation
8	Burglary alert situation
9	Dripping alert situation
10	Cleaning situation

3.3 SOFNN training and testing phases

The purpose of the SOFNN training in the cognitive reasoning module is to enable the network to build its neuronal structure based on adding and pruning neurons. The training process for the SOFNN is distributed where each sub-module (SOFNN 1 through *m*) addresses a desired reasoning output while considering all incoming inputs (input 1 to *r*). A set of rules for each output is generated through this structural development. The training phase used 2410 synthesized data samples.

3.4 Neuron Adding and Pruning

The major advantage of a SOFNN is that it can modify its neuronal structure dynamically according to the received inputs while maintaining the desired outputs. The parameter learning makes the network converge quickly through recursive least squares algorithm and the structure learning attempts to achieve an economical network. Unlike a fixed network which requires expert knowledge and a *priori* information of input-output relations, this self-organising feature enables the SOFNN to add and prune neurons as data flows through the network (Fig. 8) giving it a compact structure as described in sections 2.3 and 2.4. The network has started with one neuron for each reasoning output and finally it has 86 neurons that are essential for the desired reasoning outputs.

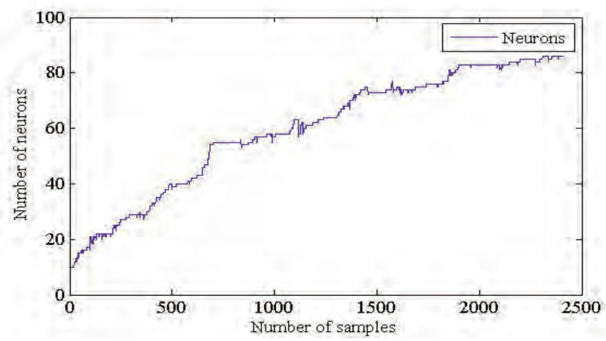

Fig. 8. SOFNN structure through adding and pruning of neurons

Each sub-module of the overall SOFNN is responsible for one reasoning objective. These modules have their own topology which exhibit growth and deletion of neurons. Fig. 9 shows the structure of the network for identification of a fire situation. The number of neurons has grown to 11 and finally settles at 3 after a number of neuron addition and pruning operations.

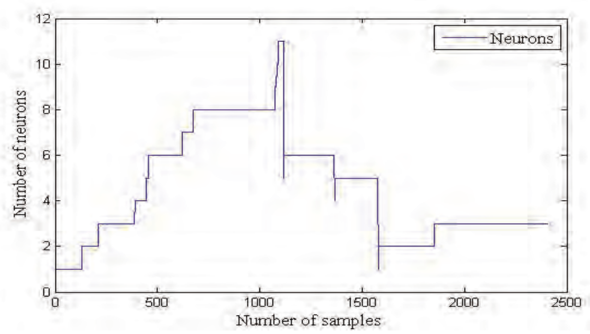

Fig. 9. Neuronal structure for the reasoning of fire situation

3.5 Extraction of Rules

A set of rules for each SOFNN module is generated through this structural development. For example, there are three neurons in the current structure which represents the fire situation objective. Fig. 10 shows the fuzzy membership functions (MFs) of input x(1). The rules generated from the fire situation objective (output 7) are as follows:

Rule 1:
If $x(1)$ is $A(0.7, 0.8291)$ and $x(2)$ is $A(0.25, 0.9628)$ and $x(3)$ is $A(0.32, 0.9751)$ and $x(4)$ is $A(0.42, 0.7792)$ and $x(5)$ is $A(0.22, 0.8665)$ and $x(6)$ is $A(0.44, 0.677)$ and $x(7)$ is $A(0.57,$

0.6957) and $x(8)$ is $A(0.84, 1.0361)$ and $x(9)$ is $A(0.57, 0.587)$ and $x(10)$ is $A(0.2, 1.1135)$ and $x(11)$ is $A(0.21, 0.9635)$ and $x(12)$ is $A(0.32, 0.784)$ and $x(13)$ is $A(0.79, 0.8989)$ and $x(14)$ is $A(0.37, 0.8507)$ and $x(15)$ is $A(0.21, 1.0603)$ and $x(16)$ is $A(0.84, 0.999)$ and $x(17)$ is $A(0.78, 0.2239)$ and $x(18)$ is $A(0.3, 0.5597)$ and $x(19)$ is $A(0.44, 0.5685)$,
then, $y(7)_1 = 0.0535-0.0044x(1)-0.0055x(2)-0.0021x(3)-0.007x(4)-0.0057x(5)-0.0018x(6)+0.033x(7)-0.0143x(8)+0.5006x(9)+0.0092x(10)-0.0282x(11)-0.0394x(12)-0.0144x(13)-0.0267x(14)-0.0015x(15)+0.4579x(16)-0.0526x(17)+0.0164x(18)-0.021x(19)$

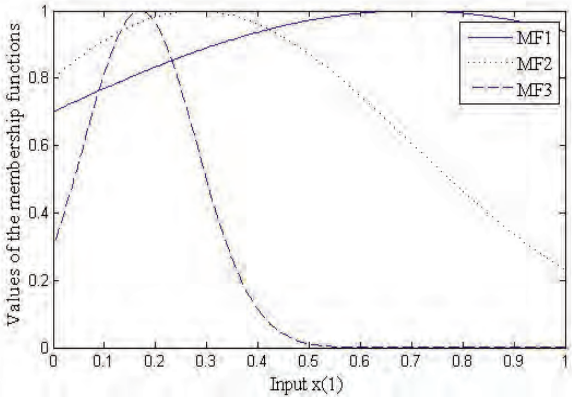

Fig. 10. Fuzzy membership of input x(1)

Rule 2:
If $x(1)$ is $A(0.28, 0.42)$ and $x(2)$ is $A(0.45, 0.2)$ and $x(3)$ is $A(0.92, 0.6)$ and $x(4)$ is $A(0.42, 0.7792)$ and $x(5)$ is $A(0.6, 0.38)$ and $x(6)$ is $A(0.22, 0.22)$ and $x(7)$ is $A(0.76, 0.19)$ and $x(8)$ is $A(0.35, 0.49)$ and $x(9)$ is $A(0.4, 0.17)$ and $x(10)$ is $A(0.86, 0.66)$ and $x(11)$ is $A(0.21, 0.9635)$ and $x(12)$ is $A(0.32, 0.784)$ and $x(13)$ is $A(0.17, 0.62)$ and $x(14)$ is $A(0.83, 0.46)$ and $x(15)$ is $A(0.76, 0.55)$ and $x(16)$ is $A(0.84, 0.999)$ and $x(17)$ is $A(0.78, 0.2239)$ and $x(18)$ is $A(0.3, 0.5597)$ and $x(19)$ is $A(0.44, 0.5685)$,
then, $y(7)_2=-0.2994-0.3439x(1)+0.3781x(2)+0.1394x(3)+0.0174x(4)+0.2372x(5)-0.5089x(6)+0.6407x(7)-0.0838x(8)-0.1810x(9)+0.2066x(10)+0.0706x(11)+0.0937x(12)-0.1522x(13)+0.1996x(14)-0.0062x(15)-0.7832x(16)+0.0524x(17)-0.3552x(18)+0.0022x(19)$

Rule 3:
If $x(1)$ is $A(0.17, 0.11)$ and $x(2)$ is $A(0.45, 0.2)$ and $x(3)$ is $A(0.32, 0.9751)$ and $x(4)$ is $A(0.42, 0.7792)$ and $x(5)$ is $A(0.42, 0.18)$ and $x(6)$ is $A(0.68, 0.24)$ and $x(7)$ is $A(0.4, 0.17)$ and $x(8)$ is $A(0.45, 0.1)$ and $x(9)$ is $A(0.57, 0.587)$ and $x(10)$ is $A(0.2, 1.1135)$ and $x(11)$ is $A(0.86, 0.65)$ and $x(12)$ is $A(0.32, 0.784)$ and $x(13)$ is $A(0.37, 0.2)$ and $x(14)$ is $A(0.37, 0.8507)$ and $x(15)$ is $A(0.37, 0.16)$ and $x(16)$ is $A(0.72, 0.12)$ and $x(17)$ is $A(0.78, 0.2239)$ and $x(18)$ is $A(0.17, 0.13)$ and $x(19)$ is $A(0.17, 0.27)$,
then, $y(7)_3=-21.8244+0.6235x(1)-26.6523x(2)+17.0053x(3)-2.7673x(4)+14.1045x(5)-24.1942x(6)+3.4548x(7)-12.7085x(8)+34.1880x(9)+11.0181x(10)+7.8665x(11)-10.227x(12)-22.1418x(13)+11.8889x(14)+9.5529x(15)+39.371x(16)-17.2805x(17)+16.0058x(18)+0.3005x(19)$

where $x(i)$, i = 1, 2 …19 represents the input to the network; $y(m)_j$, (m =7, j=1, 2, 3), represents the rule j of the reasoning output 7; $A(c,\sigma)$ represents Gaussian membership function of corresponding input with centre c and width σ. The contribution of MFs of each neuron is calculated in the layer 2 of the SOFNN. The final output from the network is achieved through the processes as described in section 2.1.

3.6 Test Scenario Example

After training, the network is tested to exercise its reasoning capabilities using unseen data employing normal and unusual situations. The purpose is to show that it can identify anomalies and present a refined understanding of the ecology. We have synthesised another set of 301 data samples for the testing phase. Fig. 11 through to Fig. 14 show the inputs for this test scenario. Fig. 15 and Fig. 16 show the resulting reasoning outputs from the network. Here room 1 refers to a location where the user has his exercising equipment, room 2 refers to an area with a sofa and room 3 refers to the kitchen area. In Fig. 11, it is observed that the user is identified in room 1 (high confidence for samples 1 to 40), but the music event (Fig. 12) and the TV usage event (Fig. 13) have very low confidences. Under these circumstances, the cognitive reasoning module decides that the user may not be exercising (Fig. 15), although he is seen in room 1. Similarly, the user is seen in the room 3 (samples 84 to 140) with high confidences (Fig. 11), but the confidence of cooking activity is very low (Fig. 13). On the other hand, there are high confidences of usage of microwave (Fig. 12) and oven (Fig. 13) during this period. So, the cognitive layer refines the understanding of cooking activity with high confidences (Fig. 15). Moreover, the network has got high confidences of the dripping event from samples 124 to 168 (Fig. 12). As the user remains in the kitchen up to sample 140, the cognitive layer does not trigger attention for the dripping situation. However, after the sample 140, the user has left the kitchen, but there are still high confidences of the dripping event. This may be a situation of unattended tap opening. So, the cognitive layer decides to trigger the dripping alert situation (Fig. 16).

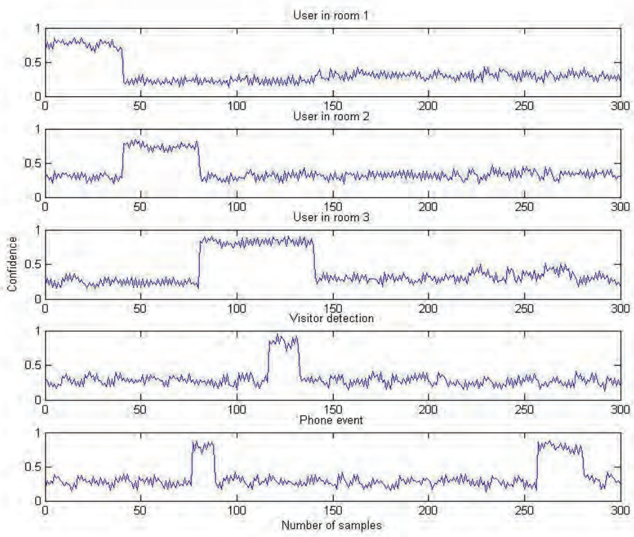

Fig. 11. SOFNN inputs 1 to 5 for test scenario

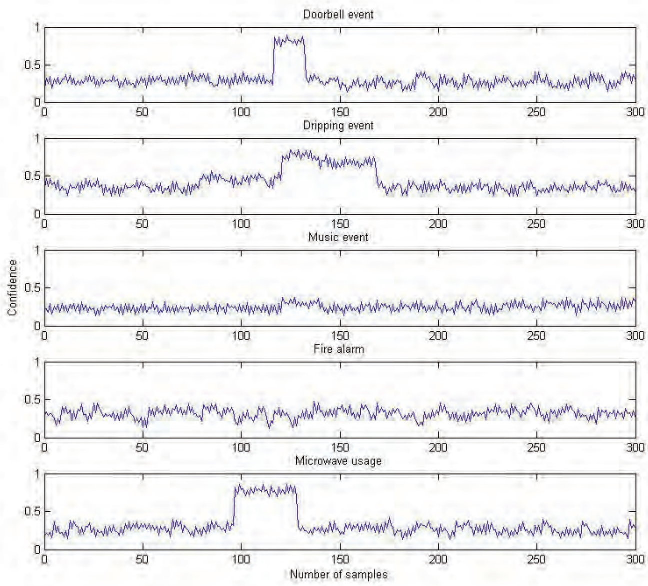

Fig. 12. SOFNN inputs 6 to 10 for test scenario

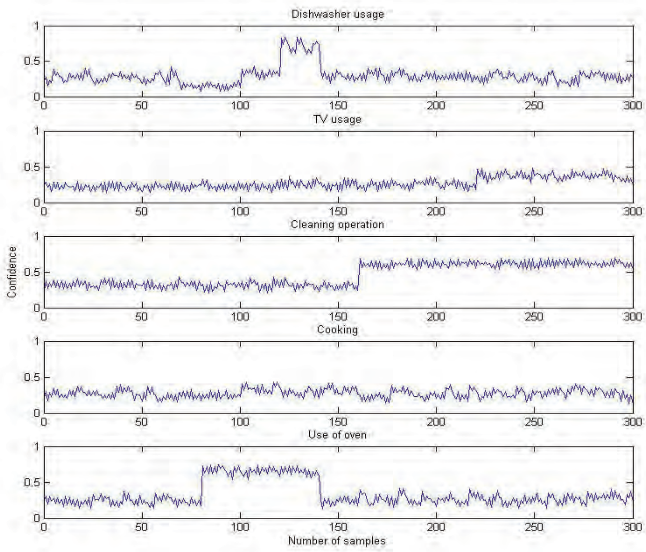

Fig. 13. SOFNN inputs 11 to 15 for test scenario

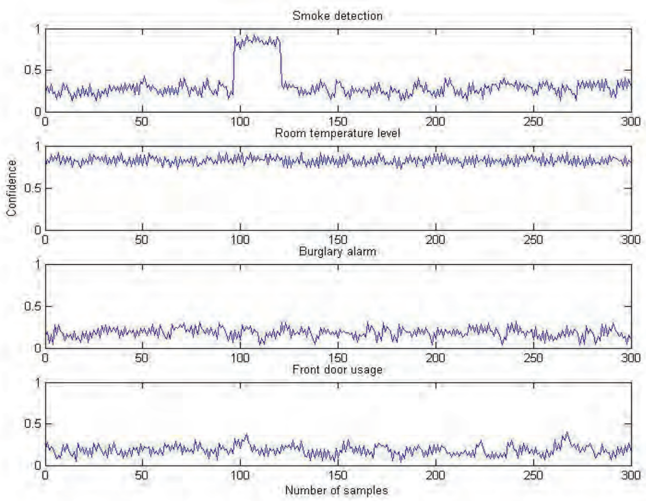

Fig. 14. SOFNN inputs 16 to 19 for test scenario

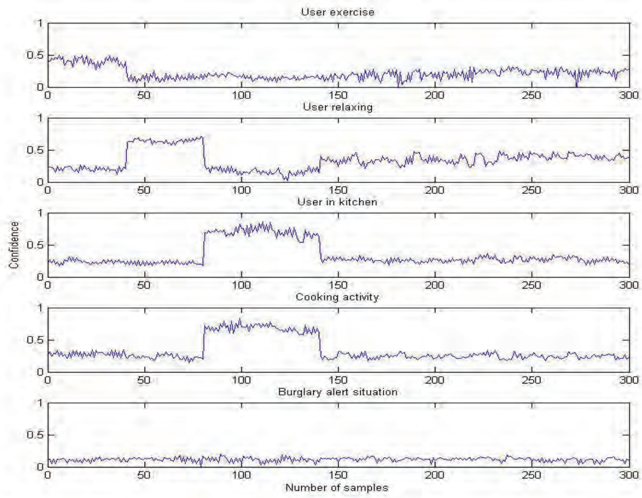

Fig. 15. SOFNN reasoning set 1 for test scenario

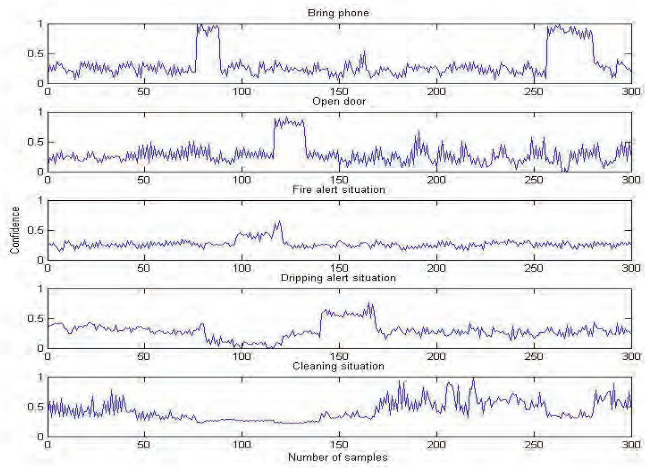

Fig. 16. SOFNN reasoning set 2 for test scenario

Another interesting point is to observe the fire alert situation. The confidences of fire alarm event are low (Fig. 12), but the network has observed a smoke event (Fig. 14) during the cooking activity. Eventually, it increases the confidences of the fire alert situation as seen in Fig. 16. So, it is observed that the developed SOFNN is capable of identifying anomalies and can refine its knowledge for the ecology.

3.7 Sliding Window Control

In this section, we present the results of neuron adding and pruning using a sliding window control based on a first in first out (FIFO) strategy. We illustrate this strategy with an example of a fire alert situation (reasoning output 7) for 2410 data samples. For demonstration purposes, we use window sizes of 100, 200 and 300 and no window control. In Table 4, we show the performances of FIFO sliding window control based on neuron number, RMSE of training and training time. Fig. 17 shows the performances of the window control methods to decide a fire alert situation expected to happen at the sample 71. It is observed that window size of 100 has the smallest training time and neuron numbers, but generates the largest RMSE. Window with 200 samples produces the minimum RMSE, but it has the largest number of neurons and the response is not suitable for test scenario as it fails to identify the fire situation with high confidence at sample 71.The other two options have similar performance for detecting fire situation. Using a window with 300 samples has two neurons and the training is faster than the method without window control; timing is an important issue while dealing with a large number of data. So, we can conclude that among the chosen window sizes, a window size of 300 samples performs better than the others to identify fire situation.

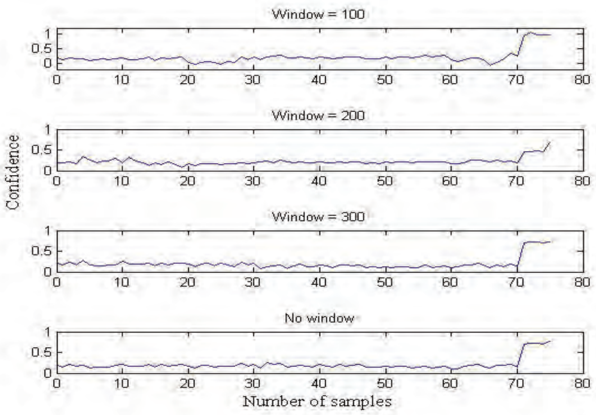

Fig. 17. Performance of various window controls for fire situation

Table 3: Performance of FIFO sliding window control for different window sizes

Window size	Neuron number	RMSE of training	Training time (s)
100	2	0.1221	30.63
200	6	0.0356	31.47
300	2	0.0840	36.25
No window	3	0.0769	48.68

4. CONCLUSIONS

The SOFNN based cognitive reasoning module has been designed, developed and implemented and utilised to extract knowledge from realistic events occurring within a smart home environment. The implementation of the SOFNN, including parameter matrix updating, adding strategy, pruning strategy, and windows control have been provided. Synthesized training data and testing data have been employed to scenario simulations. The results show the reasoning module has the ability to reflect the dynamics of the ecology and its cognitive capability. The state information may then be used to trigger decisions for robot operation in specified scenarios.

REFERENCES

Bregman D. (2010). Smart home intelligence - the ehome that learns. *International journal of smart home*, vol. 4, no.4.

Chen L., and Nugent C. (2010). Situation aware cognitive assistance in smart homes. *Journal of mobile multimedia*, vol. 6, no. 3, 263-280.

Cho K. B., and Wang B. H. (1996). Radial basis function based adaptive fuzzy systems and their applications to identification and prediction, Fuzzy Sets and Systems, vol. 83, 325-339.

Ferreira, P. M. and Ruano, A. E. (2009). Online sliding-window methods for process model adaptation. *IEEE transactions on instrumentation and Measurement*, vol. 58, no. 9, 3012-3020.

Gaddam A., Mukhopadhyay S. C., and Gupta G. S. (2011). Elder care based on cognitive sensor network. *IEEE sensors journal*, vol. 11, no. 3.

Izzeldin, H., Asirvadam, V.S., and Saad, N. (2011). Online sliding-window based for training MLP networks using advanced conjugate gradient. *IEEE 7th International colloquium on signal processing and its applications*. 112-116.

Jakkula V., and Cook D. J. (2011). Detecting anomalous sensor events in smart home data for enhancing the living experience, *AAAI workshop*, 33-37.

Jang J. S. R. (1993). Jang, Anfis: Adaptive-network-based fuzzy inference system, IEEE Transactions on Systems, Man, and Cybernetics, vol. 23, no. 3, 665-684.

Leng G., McGinnity T. M., and Prasad G. (2005). An approach for on-line extraction of fuzzy rules using a self-organising fuzzy neural network. *Fuzzy sets and systems,* vol. 150, no. 2, 211-243.

Leng G., Zeng X-J., and Keane J. A. (2012). An improved approach of self-organising fuzzy neural network based on similarity measures, Evolving Systems, vol. 3, 19–30.

Mastrogiovanni F., Sgorbissa A., and Zaccaria R. (2010). A cognitive model for recognizing human behaviours in smart homes. *Ann. Telecommunication,* vol. 65, 523–538.

Roy P. C., Giroux S., Bouchard B., Bouzouane A., Phua C., Tolstikov A., and Biswas J. (2010). Possibilistic behavior recognition in smart homes for cognitive assistance, *Twenty-fourth AAAI workshop,* 53-60.

RUBICON project. (2011). *EU FP7 project. FP7 challenge 2, cognitive systems and robotics.* Available: http://www.fp7rubicon.eu.

Son J. Y., Park J. H., Moon K. D., and Lee Y. H. (2011). Resource-aware smart home management system by constructing resource relation graph. *IEEE transactions on consumer electronics*, vol. 57, no. 3.

Takagi, T., and Sugeno M. (1985). Fuzzy identification of systems and its applications to modeling and control. *IEEE transactions on systems, man, and cybernetics*, vol. 15, no. 1, 116-132.

Wang N., Er M. J., and Meng X. (2009). A fast and accurate online self-organizing scheme for parsimonious fuzzy neural networks, *Neurocomputing,* vol. 72, no. 16-18, 3818–3829.

Wang W. Y., Chuang C. C., Lai Y. S., and Wang Y. H. (2005). A context-aware system for smart home applications. *EUC workshops*, LNCS 3823, 298 – 305.

Wu S., Er M. J., and Y. Gao. (2001). A fast approach for automatic generation of fuzzy rules by generalized dynamic fuzzy neural networks, *IEEE Transactions on Fuzzy Systems*, vol. 9, no. 4, 578-594.