Embedding a Neural Network into WSN Furniture

Symone Gomes Soares¹, Adson Ferreira da Rocha², Talles Marcelo G. de A. Barbosa³, Rui Alexandre de Matos Araújo¹

¹Systems and Robotics Institute, University of Coimbra, Coimbra, Portugal. E-mail: symonesoares@isr.uc.pt, rui@isr.uc.pt
²Department of Electrical Engineering, University of Brasília, Brasília, Brazil. E-mail: adson@unb.br
³Department of Computing, Pontifical Catholic University of Goiás, Goiânia, Brazil. E-mail: talles@ucg.br.

Abstract—Wireless Sensor Networks (WSN) is an emerging technology that is developed with a large number of useful applications. On the other hand, Artificial Neural Networks (ANN) have found many successful applications in nonlinear system and control, digital communication, pattern recognition, pattern classification, etc. There are many similarities between WSN and ANN. For example, the sensor node itself can be seen as a neuron since the WSN application show characteristics such as distributed processing, massive parallelism, adaptively, inherent contextual information processing, fault tolerance and low computation. This paper examines the possibility of embedding ANN and WSN into a Smart Table. Prototypal results have shown that ANN models are good candidates for using it deployed into low cost System-on-a-Chip (SoC).

Keywords-component: neural network; sensor network; smart home; WSN; furnure; appliance.

I. INTRODUCTION

The miniaturization and cost reduction of microelectronic devices have been leading to the development of new technologies. WSNs are one example of these new technologies. A WSN is a distributed system that is composed of autonomous units with sensing capabilities, interconnected by wireless. WSN is an emerging technology that is developed with a large number of useful applications since environmental monitoring until body sensor networks. Besides, the inherent characteristics of a WSN allow it to be easily applied inside the Smart Home or into domestic furniture (or Appliance). Nowadays, there are many applications such as a dining table [1] and smart refrigerator [7]. Besides, there are many other possibilities because all kind of furniture can be enhanced to act as an embedded computer, when a WSN is embedded into.

On the other hand, ANNs have found many applications in control, digital communication, pattern recognition and pattern classification. However, there are similarities between WSN and ANN. The sensor node itself can be seen as a neuron since the WSN application show characteristics such as distributed representation and processing, massive parallelism, adaptively, fault tolerance and low computation. Thus, traditional ANN can be easily embedded into WSN to promote new application developments such as appliances. This kind of hybridization is a promising research field of computational intelligence focusing on synergistic combinations of multiple approaches to develop the next generation of intelligent systems [10].

This paper examines the possibility of embedding ANN and WSN into an appliance called Smart Table. This kind of application is a very interesting case study because it requires at least distributed representation and processing, pattern classification. It is necessary to reach these goals to explore some aspects related to the topological organization, data fusion and programming support. These aspects are fundamentals to put into operation artificial neurons building inside sensor nodes. Besides, some observations can be useful contributions in other application areas such as Soft Sensors and their applications in industrial environment [4]. Prototypal results have shown that ANN models are good candidates particularly for using it deployed into low cost microcontroller.

The next section gives an overview of current embedded ANN into WSN applications. Section III presents a WSN application development. Section VI presents some quantitative results extracted from benchmark tests. Finally, section V discusses the results and some future perspectives for working.

II. APPLIANCES, WSN AND ANN

Appliances are intelligent artifacts that make the user home’s life more comfortable [9]. In [11] is presented a smart table, an appliance able to identify objects. Appliances may be implemented by a single processor or by the perspective of a WSN. The aim of a WSN is to design tiny autonomous computers that can unobtrusively observe their environment through built-in sensors and report back to a remote station [6]. What makes WSN interesting from a distributed system perspective is that they are used for processing information. Thereby, they do more than provide communication services, which is what traditional computer networks are all about [12]. The implementation of an appliance through a WSN has as benefits: fault tolerance and distributed processing capacity.

Another fundamental aspect in the appliances project using WSN refers to the capacity of data fusion, inherent in a WSN. Data fusion is a very ample term, and used in several studies as a formal structure in which are expressed methods and tools to data junction origination of different source. Your goal is obtaining of higher quality information, the exact definition of “higher quality” will depend of the application [13]. According [14], data fusion may be segmented at various levels of abstraction. In the lower level are basics features, such as filtering data. In the intermediary level, data are transformed into information in order to may be used in decisions making the higher level. For example, pattern recognition is a feature of intermediary level from the model proposed in [14].

Models may be used in the data fusion to transform data. These techniques can be classified into three different groups. First, fusion based on probabilistic models, second, fusion
based on least-squares techniques and, third, intelligent fusion. The probabilistic model methods are Bayesian Networks (BN), evidence theory, robust statistics and recursive operators. The least-squares techniques are Kalman filtering, optimal theory, regularization and uncertainly ellipsoids. The intelligent methods are fuzzy, ANNs and genetic algorithms [15]. “Black-box” models, such as ANN, have some advantages over probabilistic models, such as BN. First, it is not necessary any prior knowledge about the system. Second, it is computational difficult of exploring a previously unknown network. Beyond, for classification pattern, ANNs are usually adopted.

Capability of fusion, ANNs and WSN present other similar characteristics. ANNs is a model that consists of processing elements, connections and an output. The sum of all weights and a threshold determine the output. Analogically, a node sensor converts physical quantities to an electrical output signal which is filtered similarly to weighting. Hence, a sensor node can be extended in a WSN topology. A WSN can be seen as an ANN, which a sensor node inside the WSN could run an ANN model to decide on the output action.

The division of an ANN in units or modules is also known as Modular Neural Network (MNN). An ANN is said to be a modular if the computation performed by the network the network can be decomposed into two or more modules that operate on distinct inputs without communicating with each other. The benefits of creating an ANN in modules are model complexity reduction, robustness, scalability, learning capacity, economy of learning and knowledge integration [16].

Embedding a complex ANN model into a sensor gives rise some challenges, for example, energy consumption is directly proportional to computation complexity. In [3] is proposed an ANN model called Laguerre Neural Network (LaNN). The goal of this model is to reduce the complexity and energy consumption in a node. Embedding an ANN into a sensor node imposes some limitation on the architecture. However, more optimized ANN with this purpose are required, as [5].

III. EMBEDDING AN ANN INTO WSN FURNITURE

In this paper, we discuss the Smart Table as a case study. This furniture enables interacting with objects on the surface through localization and identification of shape of these objects. The mapping of objects on the table can be applied in several areas. In education, kids can match different objects on the table surface, and then, teachers can observe the learning progress of a particular child [11]. In smart home, the interacting with furniture can be seen as commands or tasks.

Smart Table allows simultaneous localization and classification of multiples objects. As shown in Figure 1, the appliance can identify how many objects are on surface beyond to classify their shapes in small and large. The process development of the application includes the construction of a ANN model incorporated to sensor network topology. Thus, our goal is to implement a sensor network architecture which supports the requirements of the Smart Table into a WSN topology. This concept maps the sensor network in neural structures instead of entities, bringing advantages in system architecture. Such as, it increases the performance, because hybridization reduces the abstraction in software; besides it reduces the time of prototyping, since the process is composed of a smaller number of activities.

![Fig. 1. Examples of interaction with the Smart Table.](image1)

![Fig. 2. System Architecture.](image2)

A. System Architecture

Figure 2 shows the system architecture which uses two layers. In this prototype, each input neural sensor is a node sensor of net and contains two neural models, a Perceptron and a MLP. Each input variable x is connects just a block of input neural sensor. The outputs y of all blocks are connected to an output block. This block is able to unite the information of each input block and generate the application states.

![Fig. 3. Overview of the proposed system.](image3)
B. Hardware

The hardware architecture of a neural sensor consists of a PIC18F4550 microcontroller and four light sensor. The PIC18F4550 is a specific purpose chip. The communication standard used for interconnected nodes is RS-232. On the other hand, ZigBee wireless standard is used to connect the appliance to a computer. ZigBee is a standard with focus on application for monitoring and sensing and may control 65535 nodes in a network with low power consumption. The light sensor is a low cost sensor which reduces electrical resistance when it captures a light energy. So that the light captured may be converted into voltage, each sensor is configured as a voltage divider. In Equation (1) shows behavior of the LDR sensor wherein $V_{\text{Saida}}$ is output voltage divisor, $R_{\text{ldr}}$ is LDR resistance and $R$ is a resistance connected in series to the sensor.

$$V_{\text{Saida}} = \frac{V_{cc} \cdot R}{R + R_{\text{ldr}}}$$  \hspace{1cm} (1)

C. Input Neural Sensor

The simplest ANN model is the Perceptron. The architecture of the Perceptron is feedforward and three-layered. The first layer works as a buffer where data are stored. The second layer, called “Feature Layer”, each neuron combines information coming from different sensory. In the last layer, “Perceptron Layer”, the weights between the buffer and the Feature Layer are fixed. Perceptron presents some problems, because it may not be used in more complex application, but only in structures of simple decision [8].

On the other hand, the most important example of ANN is the Multilayer Perceptron (MLP). The MLP can solve more complex solutions and it provides optimal solution to arbitrary classification problem and pattern recognition. The most popular method for training this model is the backpropagation algorithm, an extension of the LMS algorithm [10].

![Fig. 4. Activity Diagram of the Input Neural Sensor.](image)

Each input neural sensor contains ANN models: a Simple Perceptron and MLP. Figure 4 shows the activity diagram of the system. First, node sensor reads the light sensors connected with it. Later, a Perceptron detects the presence of objects on the node. If an object is detected, then, a MLP routine is called to classify objects. Otherwise, the node ends its activity. The purpose of using two ANN models is to reduce the processing.

D. Simple Perceptron

Simple perceptron is an ANN model used to detect objects. The development process of the Perceptron developments is divided in: ANN configuration, training, test and integration. The ANN configuration is: a neuron and a neuron layer. Sensors work as inputs for the neuron. Training is responsible to adjust connection weights in order to later classify light patterns. In this project, training is done on computer, because training on microcontroller requires more memory space. This does not affect the performance of the system, because ANN is a robust model [8], even if one of the weights is damaging on microcontroller, the system would work well.

The perceptron is trained with AND logic function. Thus, considering “true” state is the light presence. So perceptron produces “true” output when all sensors are “true”. Moreover, at least one of sensors is “false”, the perceptron produces output “false”. At the end of training, weights of each input will be returned by program on computer.

![Table 1](image)

Tests was successfully performed with two inputs. The next step is integrating. A data structure in vector, as shown in Figure 5, will receive the weights of connections between neuron and inputs (sensors). The number of positions is number of sensors ($n$) plus bias. The data integration from training requires the steps: sensor readings, conversion of readings, normalization, activation function calculation and threshold function calculation. The normalization is described through Equation (2) and (3), wherein $m_L$ is environment lighting average in a time $t$, $v$ is the digital value and $e$ is the input of perceptron. Therefore, only readings less than $m_L$ are normalized, readings above than $m_L$ are considered value 1.

$$v \geq m_L \Rightarrow e = 1$$  \hspace{1cm} (2)

$$v < m_L \Rightarrow e = v / m_L$$  \hspace{1cm} (3)

E. Multilayer Perceptron

The second structure added to the microcontroller is MLP. This model classifies patterns of objects according to size: “large” e “small”. The network configuration evaluated for the MLP is a three-layer net. This configuration can implement any function from input to output. [10]. The amount of inputs $n$ is equal to the amount of neurons in first layer and the last layer has one neuron. In an initial test scenario, smart table has four light sensors (inputs). In the training are used 16 ($2^4$) examples. The logic used to classify objects is: to three or four sensors without light, object is classified as “large”; to zero to two sensors without light, object is “small”.

The MLP training is performed on MatLab. The value of elements MLP (bias, weights, learning rate etc.) were carefully chosen to have good performance. For example, the learning rate has value 0.3, as suggested in [8] for the weights converge. And weights started between the range [-1/n; 1/n]. The weights are arranged in matrices, one matrix for each neuron layer. The number of memory needed positions in PIC to insert the matrix is described in Equation (4). Wherein $l$ corresponds to the number of MLP layers, $n_i$ is the number of neurons in layers $i$ and $E$ is the inputs of layer.

$$\sum_{i=1}^{l} n_i \cdot x \cdot (E + \text{bias})$$  \hspace{1cm} (4)

For the classification of object in the microcontroller is necessary to perform the feedforward. This is accomplished
with the multiplication of inputs by weights, from initial layer to final layer, until the production of an output.

IV. RESULTS

The goal of tests is evaluate the implementation of a neural structure in a node sensor. The first test aims to evaluate the ROM memory consumption in PIC based on the number of hidden neurons, in a three-layer net, as shown in Figure 6. As can be seen, there is a linear increase in memory consumption when the number of neurons on the microcontroller expand. The second test evaluates the time of response in microcontroller based on the number of hidden neurons, as displays in Figure 7. The time of response includes reading of sensors to response of network. This test was performed with the PIC18 Simulator IDE [2]. There is a considerable growth of the time of response when new hidden neurons are added. This happens, because data structure in node expands, it also elevates the processing and consequently time of response.

The third test examines if the output of the ANN is correct. Four sensors were connected to the microcontroller. The network classifies objects in “none” “small” and “large”, as shown in Table II. To cover the sensors were used a small object (15x15cm) and a large object (30x30cm). In the Table II, column “d” represents the distance between sensor and object. The “size of object” shows the standard of object used in the test. The “output of NN” displays the output of the ANN. “Result” shows if the network classified correctly the objects based on the size of the object used in the test. The tests display that only 7 and 9 had incorrect results. This is attributed by largest distance between sensors and objects.

In this paper, we proposed the implementation of embedding ANN into a WSN furniture, called Smart Table. This application is interesting as a case study, because it requires distributed processing and representation, pattern recognition and pattern classification, which are advantages inherent of WSN and ANN applications. The mapping and hybridization of these networks provide benefits, such as, reduction of the time of prototyping and composition of the system in a smaller number of activities. Tests show possible to integrate a chip with low memory capacity and a complex computational model. Moreover, the introduction of new inputs increases processing. Another test is performed to verify whether the outputs of the network are correct. The test shows that just two outputs are incorrect due to greater distance between sensors and object. Future works include the development of a programming language which enables the direct mapping of the functional requirements that describe an application in an ANN model and, consequently, in a WSN topology. Moreover, this hybridization model can be expanded in another models, such as, Hidden Markov Model (HMM ).

V. CONCLUSIONS

TABLE II. RESULT OF THE NEURAL NETWORK

<table>
<thead>
<tr>
<th>Covered Sensors</th>
<th>d (cm)</th>
<th>S₁</th>
<th>S₂</th>
<th>S₃</th>
<th>S₄</th>
<th>Size of Object</th>
<th>Output of NN</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>222</td>
<td>226</td>
<td>220</td>
<td>225</td>
<td>None</td>
<td>None</td>
<td>Correct</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>226</td>
<td>225</td>
<td>Small</td>
<td>Small</td>
<td>Correct</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>10</td>
<td>93</td>
<td>219</td>
<td>210</td>
<td>218</td>
<td>Small</td>
<td>Correct</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>0</td>
<td>40</td>
<td>23</td>
<td>233</td>
<td>224</td>
<td>Small</td>
<td>Correct</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>10</td>
<td>75</td>
<td>86</td>
<td>227</td>
<td>235</td>
<td>Small</td>
<td>Correct</td>
</tr>
<tr>
<td>6</td>
<td>3</td>
<td>0</td>
<td>28</td>
<td>33</td>
<td>40</td>
<td>222</td>
<td>Large</td>
<td>Correct</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>10</td>
<td>57</td>
<td>82</td>
<td>64</td>
<td>129</td>
<td>Large</td>
<td>Correct</td>
</tr>
<tr>
<td>8</td>
<td>4</td>
<td>0</td>
<td>25</td>
<td>18</td>
<td>16</td>
<td>31</td>
<td>Large</td>
<td>Correct</td>
</tr>
<tr>
<td>9</td>
<td>4</td>
<td>10</td>
<td>89</td>
<td>97</td>
<td>104</td>
<td>83</td>
<td>Large</td>
<td>Incorrect</td>
</tr>
</tbody>
</table>

REFERENCES