IMPLEMENTATION OF THE PERCEPTRON ON ARDUINO UNO DEVELOPMENT BOARD

ANDRIOAIA DRAGOS-ALEXANDRU[1], CULEA GEORGE[1], ROTAR DAN[1]

[1] *University"Vasile Alecsandri"of Bacău, No. 157 Calea Mărăşeşti Street, Bacău County , 600115, Romania*

Abstract: In recent years, researches regarding the development of new applications of neural networks have intensified. In this paper, the authors describe the way of implementation of a perceptron on a development board Arduino UNO. The perceptron training was programmed to be realized by using the graphic interface Serial Monitor included in the development environment Arduino IDE. The verification of the answer given by the perceptron was realized throughout an electric circuit.

Keywords: perceptron, single neuron, neural networks

1. INTRODUCTION

Researches regarding the electronic circuits design which could copy as accurate as possible biological neural networks and their attributes, have intensified in the last years. The main characteristic of a neural network is to learn based on prior experiences and to improve performances. The artificial neural networks are much simpler than the biological neural network, the same as the human brain the artificial neural network is composed of strong units with calculation capacity [1, 2]. The simple processing units (neurons) which compose an artificial neural network are interconnected and operate in parallel. The same as the biological neural networks, the function of the network is determined by the connection between the elements. The information inside the network is memorized based on the weights between units. The weights are adjusted in the training process so that a certain input signal could generate a certain output signal (target) [3, 4]. One can say that the neuron is the fundamental unit of an artificial neural network. A network can be trained by adjusting the weights between the units. In this paper, the authors present the means of perceptron implementation on a development Arduino UNO board.

2. THE MATHEMATICAL MODEL OF THE PERCEPTON

The simple perceptron was introduced by F. Rosenblatt in 1958 and represents a neuron. In Figure 1 is presented a simple perceptron and it represents a McCulloch-Pitts neuron provided with a learning mechanism. The represented perceptron receives signal on input X_i, $i = 1,2,3 \ldots . n.$ and emits a single signal of exit on output *Y*. The input signals can come from external environment or from outputs of other neurons. The input signals are weighted with weights W_I, which show the importance of the respective input when the output signal is generated. The two vectors used are: vector of the inputs $X = \lfloor X_0 \; X_1 \ldots . X_n \rfloor^T$ and vector of the weights $W = \lfloor W_0 \; W_1 \ldots . W_n \rfloor^T$. The weights of the input signals can have negative values being called inhibitive weights or positive values being called excitatory weights. The input *b* is the threshold entry (bias) and expresses the initial state of the perceptron [3, 4, and 5].

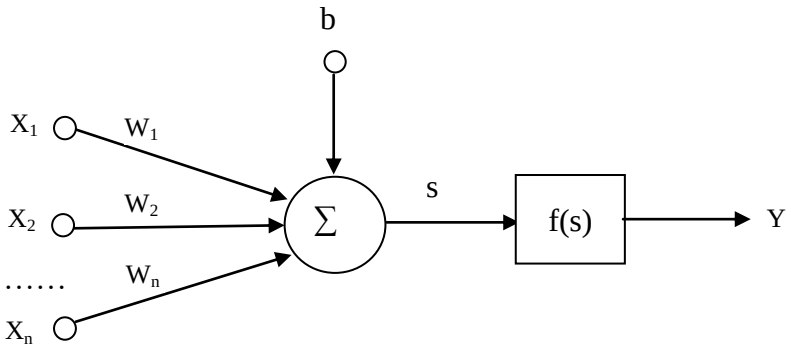

Fig. 1 Schematic representation of the perceptron

The internal state or the activity of the neuron can be represented as bellow: [5]:

$$s = \sum_{i=1}^{n} X_i W_i + \mathbf{b} \tag{1}$$

The output of a neuron depends on the internal state of the $Y = f(s)$ and on the activation function. From the most commonly used activation functions one can list the following: threshold function, ramp function, sigmoidal function, Gauss function. At the implementation of the perceptron from this paper, the binary function was used. This function reduces the output domain of the neuron at two values {0, 1}. The mathematical expression of the threshold function is [4, 6,7]:

$$f(s) = \begin{cases} 0 \; daca \; s < 0 \\ 1 \; daca \; s \geq 0 \end{cases} \tag{2}$$

From the point of view of the input sizes, one can say that this neuron can realize a classification of those into two classes: A_1 corresponding to output $Y = 1$, respectively A_2 which is obtained for 0. If $W^T X > 0 \Rightarrow Y = 1$, and if $W^T X < 0 \Rightarrow Y = 0$. The decision areas can be separated by a hyper plan given by the relation:

$$\sum_{i=1}^{n} X_i W_i + \mathbf{b} = 0 \tag{3}$$

For the case in which $n = 2$, the previous equation transforms into:

$$X_1 W_1 + X_2 W_2 + \mathbf{b} = 0 \tag{4}$$

The equation above represents the equation of a straight line (in the plan). The two classes will be separable throughout a straight line. The perceptron can be used for memorizing the simple logical functions of two variables such as AND, OR. The more complex logical operations such as XOR cannot be resolved by using a single neuron, the presented points are not linear separable. For > 3 the equation describes a hyper plan [8].
The training of a single neuron is done by reducing the values of the weights iteratively in a process of minimizing the objective function, to reduce the differences between the real output data and the wanted output data. The training data are given as [4, 5]:

$$X_d = \begin{bmatrix} X_{11} & \dots & X_{1m} \\ \dots & \dots & \dots \\ X_{n1} & \dots & X_{nm} \end{bmatrix} \text{ şi } \; T_d = [T_1 \quad \dots \quad T_m] \tag{5}$$

Where:
$d = 1,2,3 \dots . m$ – is the number of the sets of data;
T^d - is the target vector (the vector of the wanted values) for the set of input data;
The learning algorithm is based on the minimizing of an instance of the data set and supposes the taking of the following steps [4, 6, 7]:
P1. Weights initialization. The initial weights will be initialized with small values different of zero. $W_i = random(\min, [\max)]$, $i = 1,2..n$. Generally, there can be used values between the interval [-0.1, 0.1].

P2. It is established the value if the learning constant $0 < \alpha < 1$.

P3. For each data set ($d = 1,2,3 \ldots . m$):

- it will be calculated the internal state of the neuron S_d ;
- it will be calculated the real output of the neuron Y_d .

P4.For each weight W_i :

- will be calculated the value of adjustment of the weight with the relation:

$$\Delta W_i = \alpha(T_d - Y_d)X_{id} ; \quad (6)$$

- it will be adjust the weight:

$$W_i = W_i + \Delta W_i . ; \quad (7)$$

P5. Stop condition. It is repeated P3 and P4 until the values of the weights do not repeat a number of consecutive steps equal with the data set. If the learning constant is chosen at a too low value than it will result a slow process of learning and if a too high value will be chosen, the algorithm could determine an oscillating process of learning which could never conclude.

3. IMPLEMENTATION OF THE PERCEPTRON ON THE ARDUINO UNO DEVELOPMENT BOARD

The mathematical model of the perceptron was implemented on a development Arduino UNO board. The development Arduino UNO board uses for the process of the data, the microprocessor ATmega328 which has: 32 KB Flash memory, 2 KB SRAM memory și 1 KB EEPROM memory. The data is processed with a speed of 16 MHz. In order to show the availability of the model, it was used the wiring diagram from Fig. 2 [9].

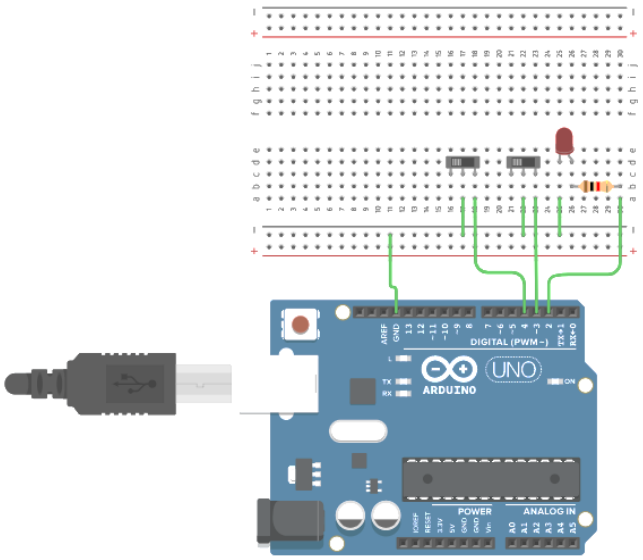

Fig. 2 Electrical wiring diagram

In the wiring diagram from Figure 2, the red led is used to visualize the answer of the perceptron according to the applied signals on the input throughout the three switches after the training was realised. The led was connected at pin 2 and the switchers at pin 3 and 4 in the digital portal. A double input neuron was used.

3. RESULTS AND DISCUSSIONS

The training of the perceptron was programmed to be done by using the graphic interface Serial Monitor, included in the development environment Arduino IDE. Throughout this interface, it can be done an exchange of information, bidirectional between the Arduino UNO board and a PC, by using the serial transmission.

The Serial Monitor interface throughout which the training of the perceptron is made is presented in Figure 3.

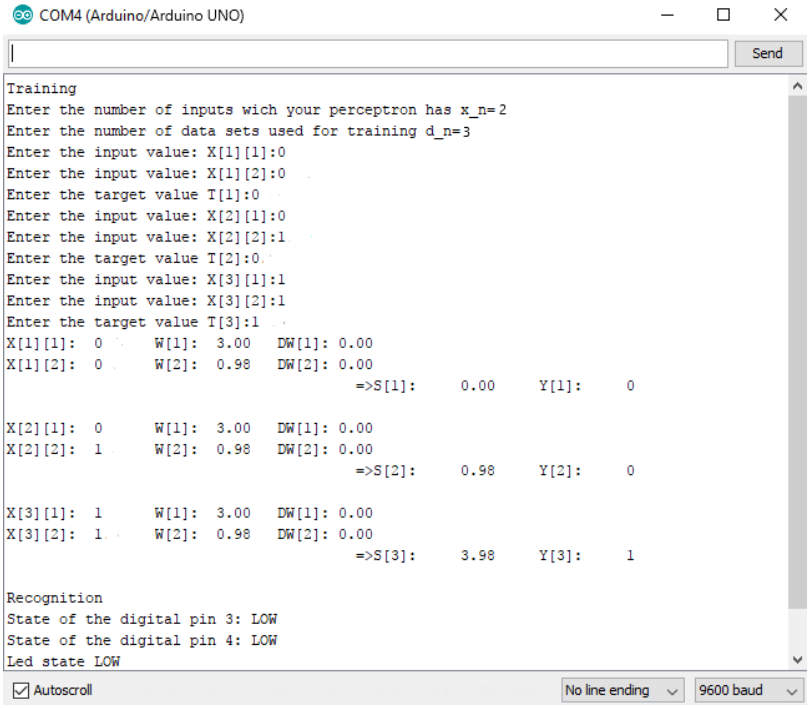

Fig. 2 Interface Serial Monitor

In the process of training, the user is asked to introduce the number of inputs of the perceptron, the input data set and also the output signal suitable for each set of the input signals. If the training could be done, than the training can be tested through the wiring diagram realized.

4. CONCLUSIONS

In the paper, it was implemented a single neuron on the development Arduino UNO board. It was used 10% of the program memory. In order to develop a more complex neural network is required the usage of a development board which contains a microprocessor with a higher memory capacity.

REFERENCES

[1] Zhang X. S., Introduction to Artificial Neural Network, ISBN 978-1-4419-4836-6, Ed. Springer, Boston, MA. 2000.
[2] Tiponut V, Caleanu C. D., Rețele neuronale, Ed. Politehnica Timisoara, 2002.
[3] http://www.cs.ubbcluj.ro/~lauras/test/docs/school/IA/lectures2013/lectures/09_10_ML_ANN.pdf (25.07.2017).
[4] http://hagan.okstate.edu/4_Perceptron.pdf (25.07.2017).
[5] Rotar D., Inteligență artificială, Ed. Alma Mater, Bacău, 2017.
[6] Samantak B., Al-Balushi R., Artificial neural network based fault diagnostics of rolling element bearings using time-domain features, Mechanical Systems and Signal Processing, vol 17, no 2, 2003, p. 317-328.
[7] Subana S., Sandhya S., Artificial Neural Network Modelling, ISBN 978-3-319-28493-4, Ed. Springer, Cham, 2016.
[8] Jinde C., Jinling L., Boundedness and stability for Cohen–Grossberg neural network with time-varying delays, Journal of Mathematical Analysis and Applications, vol. 296, no. 2, 2004, p. 665-685.
[9] https://www.arduino.cc/ (25.07.2017).