Design and implementation hamming neural network with VHDL

Liqa Saadi Mezher
Department of Computer Engineering Al-Mustansiriyyah University, Baghdad, Iraq

ABSTRACT

Hamming Neural Network is type of artificial neural network consist of two types of layers (Feed Forward Layers and Recurrent Layer). In this paper, two inputs of patterns in binary number were used. In the first layer, two neurons and pure line function were used. In the second layer, three neurons and positive line function were used. Also applied Hamming Neural networks algorithm in three simulation methods (Logical gate method, software program coding method and instant block diagram method). In this work in VHDL software program was used and FPGA hardware used.

1. INTRODUCTION

Artificial neural networks (ANN) is a technique of artificial intelligence inspired by the human neurons system and is commonly used for modeling and improving the complex phenomena that involve a large number of process variables [1-3]. Artificial neural networks are computational methodologies that result in multiple analyzes factors [4, 5]. Inspired by biological nervous networks, artificial neural network containing layers of simple computing nodes that operate as organs summary nonlinear models of networks [6]. Field Programmable Gate Arrays (FPGA) device is a hardware implementation large number of artificial neural network [7-9]. Very High-Speed Integrated Circuits Hardware Description Language (VHDL) the software program [10] applied to humming neural network using many methods [8, 11-13].

The paper is arranged as in the following. In the Section 2 and displays a description briefly Hamming Neural Network about the parameters of patterns, weights matrix and the transfer function using (pure line in Feed Forward layer and positive line in Recurrent Layers). In Section 3, displays the algorithm of Hamming Neural Network. In Section 4, Artificial Neural Network design using Hamming Neural Network with many simulation methods displays applied by VHDL software program and FPGA hardware. In Section 5, displays the result of Hamming Neural Network, last Section 6, are given some important conclusions and future work.

2. HAMMING NEURAL NETWORK

In this paper, the initial parameters of the humming neural network used, the architecture of neural network contains of many layers or many neural networks, as shown in Figure 1.

2.1. Feed forward layer

In the Feed Forward Layer FFL (Feed Forward neural networks) consist of two layers [14-16] also this layer two inputs were used with one output. Where the input of binary number, the first layer (input layer of feed forward): its takes the inputs and passes them to the input layer without doing anything.
The second layer (the output layer of feed forward): an output layer, composed by n neurons, which processing n inputs, P1,…, Pn, and multiply with the weights matrix and composed by two neuron and combined with the bias and then entered into a linear activation function (Purelin (P)) and extracted the result (a1), in the FFL doesn’t have the hidden layers, as shown in Figure 1(a).

2.2. Recurrent layer

In the Recurrent Layer RCL (recurrent layer neural networks) consist of three layers [17-20], the first layer (input layer of Recurrent Layer): In this layer the output from the previous layer (feed forward) was used and used as a primitive input with one output (initial a2=0). The second layer (the hidden layer of recurrent layer): the layer of feed forward became input to the hidden layer in recurrent layer, composed by n neurons [21], which processing n inputs, P1,…,Pn, and multiply with the weights matrix and composed by three neurons and combined with the bias (b2) and then entered into a positive linear activation function (Poslin (P)). And the third layer (the output layer of recurrent layer): the result (F) was extracted, as shown in Figure 1(b).

Figure 1. Block diagram of humming neural network (a) feed forward layer (b) recurrent layer

3. ALGORITHM OF HAMMING NEURAL NETWORK

Step 1: Set activation and initialize weights and bias values.
I: Number of input nodes (input vector of Patterns).
J: Number of length of Patterns.

Step 2: For each input vector P, N=1,2, …, I, M=1, 2…, J, do step (3 to 4).

Step 3: For each neuron, multiply each input by its corresponding weight and sum with bias values. It receives for the next step.

Step 4: Update the activation for N=1,2, …, I, M=1, 2, …, J.

Step 5: Apply activation function and save current activations in Xm (old) to be used in the next iteration for M=1, 2, …, j.

\[ X_m(0) = f_{hardlim}(X_{(ni,m)}) \]

Step 6: choose an abs for the weight matrix (set 0 <absolute < 1/J).

Weight: \[ W_{nm} = 1 \quad for \quad n = m \]
\[ -\epsilon \quad otherwise \]

Step 7: For t=1,2,3, … repeat steps (8 to 11), while stopping condition is false.

Step 8: For each neuron, N=1,2,3, …, I, M=1, 2, …, J, compute the net signal, it receives for the next step.

\[ X_{(ni,m)}(new) = W_{ni,m} * P_n + B_n \]
Step 9: Update the activations for \( N=1,2, \ldots, I, M=1,2, \ldots, J \):

\[
x_m^{(\text{new})} = F_{\text{poslin}}(X_{(n,i,m)}^{(\text{old})})
\]

Step 10: Save the activation for use in the next iteration.

\[
x_m^{(\text{old})} = x_m^{(\text{new})}
\]

Step 11: Test stopping condition. If more than one node has a nonzero output then go to step 8, otherwise, as shown in Figure 2.

Figure 2. Architecture of humming neural network

4. SIMULATION METHODS FOR HUMMING NEURAL NETWORK

In this paper, 3 methods for programming were used using the VHDL program and using the FPGA device as shown below:

4.1. Logical gates method

The network is synthesized by connecting it using the NAND gate and inputs (P1, P2, W11, W12, W21, W22, W23, b1, b2) and outputs (F) are used, as shown in Figure 3.

Figure 3. Hamming neural network using logic gate
4.2. Software program coding method

In this way the programming was used by code and divided the hamming neural network into layers:

4.2.1. Feed forward layer (FFL)

In this layer, programming was done using the entity to define the inputs (P1, P2, W11, W12, b1) and output (a1), and the Logic Function was used in the architecture to extract the output as in Figure 4.

Summary: A summary review of program implementation and the special ratios of the network (FFL), as shown in Figure 5.

RTU Viewer: To review the FFL network binding after execution by (select program – tools – Netlist viewers – RTL viewer), as in Figure 6.
Design and implementation of a Hamming neural network with VHDL

(Liqaa Saadi Mezher)

4.2.2. Recurrent Neural Network (RCL)

In this layer, programming was done using the entity to define the inputs (a1, a2, W21, W22, W23, b2) and output (F), and the Logic Function was used in the architecture to extract the output as in Figure 8.

Figure 6. RTL viewer for feed forward layer

Instant Block Diagram: To view the network binding after execution the form Block Diagram is done by (select program – tools – Netlist viewers – Technology Map Viewer – Post Mapping), as in Figure 7.

Figure 7. Instant for feed forward layer

Summary: A summary review of program implementation and the special ratios of the network (RCL), as shown in Figure 9.
RTU Viewer: To review the RCL network binding after execution by (select program – tools – Netlist viewers – RTL viewer), as in Figure 10.

Instant Block Diagram: To view the RCL network binding after execution the form Block Diagram is done by (select program – tools – Netlist viewers – Technology Map Viewer – Post Mapping), as in Figure 11.
4.2.3. Hamming Neural Network (HNN)

It is the main program of the hamming neural network where the programs [22] are called for the two layers (FFL & RCL) by the component and then merge them and define the input and output and the use of the structure in architecture and extraction of output [23-25], as in Figure 12. And then convert from a software programmable to Block Diagram and implement it on the FPGA machine as a as a practical and extract the results.

```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;

LIBRARY work;

ENTITY hnnns IS
PORT
  p1 : IN STD_LOGIC;
  p2 : IN STD_LOGIC;
  w1 : IN STD_LOGIC;
  w2 : IN STD_LOGIC;
  b1 : IN STD_LOGIC;
  w1 : IN STD_LOGIC;
  w2 : IN STD_LOGIC;
  w3 : IN STD_LOGIC;
  b2 : IN STD_LOGIC;
  a1 : OUT STD_LOGIC;
  f : OUT STD_LOGIC;
END hnnns;

ARCHITECTURE bdf_type OF hnnns IS

COMPONENT ffwls
PORT(p1 : IN STD_LOGIC;
p2 : IN STD_LOGIC;
w1 : IN STD_LOGIC;
w2 : IN STD_LOGIC;
b1 : IN STD_LOGIC;
   a1 : OUT STD_LOGIC)
END COMPONENT;

COMPONENT rcls
PORT(a1 : IN STD_LOGIC;
   a2 : IN STD_LOGIC;
w1 : IN STD_LOGIC;
w2 : IN STD_LOGIC;
w3 : IN STD_LOGIC;
b2 : IN STD_LOGIC;
   f : OUT STD_LOGIC)
END COMPONENT;

BEGIN
   a1 <= SYNTHESIZED_WIRE_0;
   f <= SYNTHESIZED_WIRE_1;
   bfw_inst : ffwls
   PORT MAP(p1 => p1,
   p2 => p2,
   w1 => w1,
   w2 => w2,
   b1 => b1,
   a1 => SYNTHESIZED_WIRE_0);
   bfw_inst1 : rcls
   PORT MAP(a1 => SYNTHESIZED_WIRE_0,
   a2 => SYNTHESIZED_WIRE_1,
   w1 => w1,
   w2 => w2,
   w3 => w3,
   b2 => b2,
   f => SYNTHESIZED_WIRE_1);
END bdf_types;
```

Figure 12. Represent coding for hamming neural network
4.3. Instant block diagram method

After completing the programming process for each layer separately, take the following steps to convert the software program to block Diagram.

4.3.1. The first layer FFL

Select all software program of FFL - File - Creat/Up date - Create Symbol File for Current File, and open new file - block Diagram/Schematic File – Ok – then choose instant block from project, as shown in Figure 13.

![Figure 13. Represent instant block diagram for FFL](image)

4.3.2. The second layer RCL

Select all software program of RCL - File - Creat/Up date - Create Symbol File for Current File, and open new file - block Diagram/Schematic File – Ok – then choose instant block from project, as shown in Figure 14.

![Figure 14. Represent instant block diagram for RCL](image)

In Figure 15, the previous two instant block diagrams are integrated, and integrated network, the main input representation, the weight matrix and its main output.
In the FPGA device, 9 pins were used (9) inputs representing (2 patterns +5 weights + 2 biases) as shown Table 1 represents the used pins. And two outputs were used. One of which represents is the extraction of the interlayer from the first layer and the other represents the final output of the main network as shown Table 2 represents the used pins.

### Table 1. Represent the input pins in hamming neural network

<table>
<thead>
<tr>
<th>Switch</th>
<th>Pins</th>
<th>Represented</th>
</tr>
</thead>
<tbody>
<tr>
<td>SW0</td>
<td>PINS_L22</td>
<td>B1</td>
</tr>
<tr>
<td>SW1</td>
<td>PINS_L21</td>
<td>B2</td>
</tr>
<tr>
<td>SW2</td>
<td>PINS_M22</td>
<td>P1</td>
</tr>
<tr>
<td>SW3</td>
<td>PINS_V12</td>
<td>W11</td>
</tr>
<tr>
<td>SW4</td>
<td>PINS_W12</td>
<td>P2</td>
</tr>
<tr>
<td>SW5</td>
<td>PINS_U12</td>
<td>W12</td>
</tr>
<tr>
<td>SW6</td>
<td>PINS_U11</td>
<td>W21</td>
</tr>
<tr>
<td>SW7</td>
<td>PINS_M2</td>
<td>W22</td>
</tr>
<tr>
<td>SW8</td>
<td>PINS_M1</td>
<td>W23</td>
</tr>
</tbody>
</table>

### Table 2. Represent the output pins in hamming neural network

<table>
<thead>
<tr>
<th>LED</th>
<th>Pins</th>
<th>Represented</th>
</tr>
</thead>
<tbody>
<tr>
<td>LEDR1</td>
<td>PINS_R19</td>
<td>A1</td>
</tr>
<tr>
<td>LEDR0</td>
<td>PINS_R20</td>
<td>F</td>
</tr>
</tbody>
</table>

### 5. THE RESULT

#### 5.1. The input

The Input values for patterns and weights matrix in layers FFL and RCL, as shown Table 3.

### Table 3. Represent the input values in hamming neural network

<table>
<thead>
<tr>
<th>Patterns</th>
<th>Weights Matrix in FFL</th>
<th>Weights Matrix in RCL</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1 P2</td>
<td>W11 W12 W21 W22 W23</td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 0 0 0</td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>0 1 1 0 0</td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>1 0 0 1 0</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1 1 1 0</td>
<td></td>
</tr>
<tr>
<td>0 0</td>
<td>0 0 0 0 1</td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>1 1 1 0 1</td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>0 1 0 1 1</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>1 1 1 1 1</td>
<td></td>
</tr>
</tbody>
</table>

#### 5.2. The output

- when b1 and b2=0
  
  \[a1=11111111\]
  
  \[F=11111111\]

- when b1=1 and b2=0
  
  \[a1=00010001\]
  
  \[F=11111111\]

- when b1=0 and b2=1

---

*Design and implementation hamming neural network with VHDL (Liqaa Saadi Mezher)*
In this paper, a hamming neural network applied. This network contains two types of network FFL & RCL. In FFL layer two patterns (P1, P2) used with one output (a1). Where the input of patterns binary number was multiplied with the weight’s matrix (w11, w12) and combined with the baise (b1) and then entered into the pure line functions. In the RCL layer, the output from the previous layer (feed forward) was used and used as a primitive input (a1 and initial feedback a2=0) with one output (F). Where the input was multiplied with the weight matrix (W12, W22 and W23) and combined with the baise (b2) and then entered into the pos line function. And three simulation methods (Logical gate method, software program coding method and instant block diagram method) applied using VHDL program and connected to FPGA hardware to extract the final result.

REFERENCES


BIOGRAPHY OF AUTHOR

Liqaa Saadi Mezher, Master Sc. in Computer Engineering, B. Sc. in Computer & Software Engineering. She has 14 years of teaching experience. She published 6 research papers at International level. Worked in Al-Mustansiriyyah University, Baghdad, Iraq.