# t-Error Correcting/ d-Error Detecting ( d > t ) and All Unidirectional Error Detecting Codes with Neural Network (Part II)

*Maung Maung Htay*
*Department of Information Technology*
*Radford University, Radford, VA 24142*
*S. Sitharama Iyengar,*
*Department of Computer Science*
*Louisiana State University, Baton Rouge, LA 70803*
*and Si Qing Zheng*
*Department of Computer Science*
*University of Texas, Dallas, TX 75083*

**Abstract**

*In this paper, we develop an algorithm for t-error correcting/d-error detecting and all unidirectional error detecting (t-EC/d-ED/AUED) codes in the framework of neural work. As t-EC/d-ED/AUED codewords are formed by concatenation of information bits and one or more groups of check bits depending on how we want to construct code, we demonstrate neural network algorithms for constructing, detecting and correcting codes. As a continuation of the previous paper[25], we present an algorithm, code construction, detecting and correcting for 2EC/5ED/AUED code II. We also plan to present for other codes with more examples in near future as well.*

Keywords Neural networks, error-detecting, error correcting, checking bits, t-EC/d-ED/AUED

## 1. Introduction

In this paper, we will describe the construction of neural networks for t-Error Correcting (t-EC)/d- Error Detecting(d-ED) (d>t) and All Unidirectional Error Detecting Codes(AUED). This paper is organized as follows. In section 2, preliminary information of systematic t-EC/d-ED/AUED codes with d > t are given. Then we also describe the necessary and sufficient conditions for a code to be t-EC/d-ED/AUED with d>t and examples of code construction for code II. In section 3, we present our proposed networks for code construction and error detection/correction with the illustrative example.

## 2. Design of Systematic t-EC/d-ED/AUED Codes with d>t

In this section, we introduce the brief background information of theory and design of t-error correcting/d-error detecting (d>t) and all unidirectional error detection codes. The design of various forms of t-EC/AUED codes appear in [1][3][4][5][6][9][11][15][16][17][18][19] [20][22]. "Recently, D.J. Lin and B. Bose[11] have designed t-EC/d-UED codes with d > t for the case where the number of unidirectional effors, even though large, is limited"[14]. Since t-EC/d-ED/AUED with d > t codes designed in [14] are much more reliable than other codes with respect to redundancy, speed of encoding and decoding, and cost of implementation, we construct neural networks for detecting/correcting those codes. In the following sections, we state the background information related to the design and theory of code construction of Dimitris Nikolos [14].

### 2.1 Necessary And Sufficient Conditions

We refer definitions and theorems from [2][5][6][7][8][[13][14][17] as background information concerning our proposed model for t-EC/d-ED/AUED codes within the framework of neural network. Some definitions and theorems have been mentioned in [23][24] and our previous paper [25].

Based on the theorems 1, 2 and 3 from [25], we can construct t-EC/d-ED/AUED codes with neural networks. Before constructing a network, a method for constructing systematic t-EC/d-ED/AUED codes with d > t from [14] will be described in the next section.

### 2.2 Code Construction

Let F be a systematic t-error correcting and d-error detecting parity check code with length $n'$. Also, let $A_1, A_2,...,A_k$ with $1 \le k \le t+1$ be codes with lengths $r_1, r_2, ..., r_k$ and asymmetric distances $\Delta_1, \Delta_2, ..., \Delta_k$ such that $\sum_{f=1}^{k} \Delta_f = t+1$.

There will be two possible cases at this point for a code $A_j$; $\Delta_j = 1$ and $\Delta_j > 1$.

*Case I:*

For any code $A_j$ with $\Delta_j = 1$, we will use the binary representation of the numbers $0,1,2,...,2^{|\log(n'+1)|-a_j}$ -1;

where the value of $a_j$ is such that $2^{a_j} \le d-t+1+2.\sum_{f=1}^{j-1}\Delta_f < 2^{a_j+1}$; as a row in the matrix $M_j$, $|A_j|$ x $r_j$. That is, row m is the binary representation of m in the matrix $M_j$.

Then the cardinality of the code will be $|A_j| = 2^{|\log(n'+1)|-a_j}$ and its length $r_j = |\log(n'+1)| - a_j$.

*Case II:*

For any code $A_i$ with $\Delta_i > 1$, the rows of the matrix $M_i$, $|A_i|$ x $r_i$ are the codewords in the order of nondescending weights W(X), where X is a row in the matrix $M_i$.

The cardinality of $A_i, |A_i| \ge |(n'+1)/(d-t+1+2.\sum_{f=1}^{j=1}\Delta f)|$.

The codewords will have the form $X R_{1,i_1} R_{2,i_2} ... R_{k,i_k}$

i.e., each codeword is the concatenation of $X, R_{1,i1}, R_{2,i2}, ... R_{k,ik}$. $X$ represents the encoding of the given information bits.

For $\Delta_j \ne 1, R_{j,i_j}$ is the row $i_j$ of matrix $M_j$ with $i_j = \left\lfloor L(X)/(d-t+1+2.\sum_{f=1}^{j-1}\Delta_f) \right\rfloor$ where *L(X)* denotes the number of 0's in X.

For $\Delta_j = 1, R_{j,i_j}$ is the row $i_j$ of the matrix $M_j$ with $i_j = \lfloor L(X)/2^{a_j} \rfloor$, where the value of $a_j$ satisfies the relation $2^{a_j} \le d-t+1+2.\sum_{f=1}^{j-1}\Delta_f < 2^{a_j+1}$.

For example, let us assume that *t = 2*, and *d = 8*. According to the technique described in the above section, there will be four different 2-EC/8-ED/AUED codes as shown below.

One code contains codewords of the form $XR_{1,j_1}$ where $R_{1,i_1} \in A_1$ and $\Delta_1 = 3$.

Two codes contain codewords of the form $XR_{1,i_1} R_{2,j_2}$ where $R_{1,i_1} \in A_1, R_{2,i_2} \in A_2$ and $\Delta_1 = 1, \Delta_2 = 2$ or $\Delta_1 = 2$ and $\Delta_2 = 1$.

One code contains codewords of the form $XR_{1,i_1} R_{2,i_2} R_{3,i_3}$ where $R_{j,i_j} \in A_j$ and $\Delta_j = 1$ for j=1,2,3.

Similarly, for *t = 3* and *d = 8*, we will have eight different 3-EC/8-ED/AUED codes.

One code contains codewords of the form $XR_{1,i_1}$ where , and $R_{1,i_1} \in A_1$, $\Delta_1 = 4$.

Three codes contain codewords of the form $XR_{1,i_1} R_{2,i_2}$ where $R_{1,i_1} \in A_1, R_{2,i_2} \in A_2$ and $\Delta_1 = 1$, $\Delta_2 = 3$ or $\Delta_1 = 3$, $\Delta_2 = 1$ or $\Delta_1 = \Delta_2 = 2$.

Three codes contain codewords of the form $XR_{1,i_1} R_{2,i_2} R_{3,j_3}$ where $R_{1,i_1} \in A_1, R_{2,i_2} \in A_2$, $R_{3,i_3} \in A_3$, and $\Delta_1 = \Delta_2 = 1$, $\Delta_3 = 2$ or $\Delta_1 = 1$, $\Delta_2 = 2$, $\Delta_3 = 1$ or $\Delta_1 = 2, \Delta_2 = \Delta_3 = 1$.

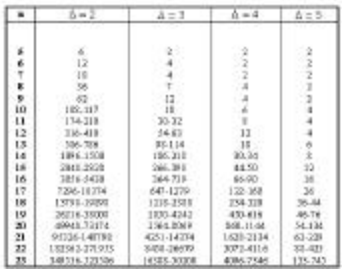
One code contains codewords of the form $XR_{1,i_1} R_{2,i_2} R_{3,i_3} R_{4,i_4}$ where , $R_{j,i_j} \in A_j$, and $\Delta_j = 1$ for j = 1, *2, 3,4.*

In the above forms of codewords, some are concatenation of information bits and one group of check bits and some are concatenation of information bits and more than one group of check bits. The main idea for all the codewords is that they are the forms of concatenation of information part and check symbol part.

### 2.3 Examples of Code

In this section, we will show examples of code construction of t-EC/dED/AUED, which are adopted from [14]. First of all, we must have a code *F* to use as a systematic parity check code. At this point, we assume that *F* has codewords of length 16 bits and for all distinct $X, Y \in F, D(X,Y) \ge 8$. According to the method shown above we can construct 2-EC/5-ED/AUED codes. In the following example codes, we use Table 1 adopted from [14][21] for the purpose of determining the length of codes, $A_j$.

**Table 1**
**Bounds Of the Cardinality of Asymmetric Error Correcting Codes**



#### 2.3.1 Algorithm for Code II

Like we have seen an example for Code I in Section 2.3.1 of [25], we construct another 2-EC/5-ED/AUED code with the codewords of the form $XR_{1,i_1} R_{2,i_2}$ where $R_{1,i_1} \in A_1$, and $R_{2,i_2} \in A_2$, with $\Delta_1 = 1$ and $\Delta_2 = 2$. Since $\Delta_1 = 1$, the cardinality of $A_1, |A_1| = 2^{|\log(n'+1)|-a_1}$, where the value of $a_1$ is such that $2^{a_1} \le d-t+1 < 2^{a_1+1}$. Thus, $a_1 = 2$ and $|A_1| = 8$. According to the method described in Section 2.2, the matrix

$M_1$ will be

$$M_1 = \begin{bmatrix} 0\,0\,0 \\ 0\,0\,1 \\ 0\,1\,0 \\ 0\,1\,1 \\ 1\,0\,0 \\ 1\,0\,1 \\ 1\,1\,0 \\ 1\,1\,1 \end{bmatrix}$$

We still need to compute the cardinality of $A_2$. According to the method shown in Section 2.2, $|A_2| \geq |(n'+1)(d-t+1+2)| = 3$. So there will be three codewords and length of each codeword is equal to 4. Then we will get the matrix $M_2$ as follows:

$$M_2 = \begin{bmatrix} 0\,0\,0\,0 \\ 0\,0\,1\,1 \\ 1\,1\,1\,1 \end{bmatrix}$$

Now-we present all the codewords of a 2-EC/5-ED/AUED code with the form $XR_{1,i_1}R_{2,i_2}$, where $R_{1,i_1} \in A_1$ and $R_{2,i_2} \in A_2$, with $\Delta_1 = 1$ and $\Delta_2 = 2$ in Table 2.

**Table 2 2-EC/5-ED/AUED Code II**

| | X | $R_{1,k}$ | $R_{2,k}$ |
|---|---|---|---|
| 001 | 1111110000001 | 010 | 0011 |
| 010 | 0001111110001 | 010 | 0011 |
| 100 | 0000001111111 | 010 | 0011 |
| 000 | 0000000000000 | 100 | 1111 |
| 011 | 1110001110000 | 010 | 0011 |
| 101 | 1111111111110 | 000 | 0000 |
| 110 | 0001110001110 | 010 | 0011 |
| 111 | 1110000001111 | 001 | 0011 |

## 2.4 A Scheme for Error Detection/Correction

In this section, we will present an error detection/correction algorithm for the tEC/d-ED/AUED codes, which is given by Dimitris Nikolos[14]. We will describe the network construction of this algorithm in a later section.

Let $Q = XR_{1,i_1}R_{2,i_2}\cdots R_{k,i_k}$ be an error-free codeword of an t-EC/d-ED/AUED code and $Q' = X'R'_{1,i_1}R'_{2,i_2}\cdots R'_{k,i_k}$ be the received codeword which has some errors.

The formal algorithm [14] for error detecting and correcting t-EC/d-ED/AUED codes is as follows:

**begin**

Let H be the parity check matrix corresponding to the systematic parity check code F, where F has been defined in Section 2.

Compute syndrome $S$ *of* $X'$, that is, $S = H.X'^T$.

Let $S$ correspond to g multiplicity error.

**if** g > $t$ **then** the error is only detectable and stop

**else** correct $X'$ using the correcting procedure in the parity check code obtaining, $X''$ as the resulting, word.

Compute $R''_{j,i_j}$, for j = 1,2.... k corresponding to $X''$

Let $Q' = X''R''_{1i_1}R''_{2i_2}....R''_{ki_k}$.

**if** $d(Q',Q'') \leq t$ **then** $Q''$ is a correct codeword

**else** errors are only detectable i.e. errors > $t$ occurred.

**End**

**Figure 1** Formal Algorithm of Error Detection/Correction

### 2.4.1 Example

In this section, we will show how error detection/correction works according to the above algorithm. In this example [14], 2-EC/5-ED/AUED Code II shown in Section 2.3.1 is assumed to be the given code. Let the parity check matrix H be

$$H = \begin{bmatrix} 1\,1\,1\,1\,1\,1\,1\,1\,0\,0\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,1\,0\,1\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0\,0\,0\,1\,1\,1\,1\,1\,0\,0 \\ 1\,1\,0\,0\,0\,1\,1\,0\,0\,1\,1\,1\,1\,1\,0\,0 \\ 1\,1\,0\,0\,0\,0\,1\,0\,0\,1\,1\,1\,1\,1\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,1\,0\,1\,1\,1\,1\,1\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0 \\ 1\,1\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,0\,1 \end{bmatrix}$$

Suppose a correct codeword

Q = 100 0000001111111 010 0011

$X$ $R_{1,i_1}$ $R_{2,i_2}$

Then we assume that the receive word would be

Q' = 101 0000001111111 101 1011

which can be checked using Table 2-EC/5-ED/AUED Code II.

According to the algorithm, the syndrome $S = H.X'^T = [0001000100111111]^T$. Since the value of syndrome S is equal to the first column of the parity check matrix H, a single error has occurred in the third position of X' in the received word Q'. Then after correcting the error, the resulting word X" = (1000000001111111).

Then we will have the new check bits $R''_{1,j_1}$ and $R''_{2,j_2}$ corresponding to X" using the formula $i_1 = \left\lfloor L(X)/(d-t+1+2\sum_{f=1}^{j-1}\Delta_f) \right\rfloor$=2. Thus $R''_{1,i_1}$ = (010) and $i_2 = \left\lfloor L(X)/(d-t+1+2\sum_{f=1}^{j-1}\Delta_f) \right\rfloor$=3. Thus $R''_{2,i_2}$ = (0011).

So, Q" = 100 0000001111111 010 0011

In order to check whether the resulting word is the correct word or not, we need to compute the Hamming distance between Q' and Q".

$$d(Q',Q'') = 2 \leq t = 2.$$

So we can say that Q" is the correct word.

## 3. The Neural Network

In this section, we present neural networks for code construction and error detection/correction. Having described code construction and error detection/correction in the above section, we will proceed to construct networks according to the described methods. Though the code constructing method in Section 2 is complicated, our proposed network construction is simple and easy to understand. Mostly the network constructions in our error correction

paradigm are similar to each other.

### 3.1 Code Construction and Compilation

As we have seen the method of the t-EC/d-ED/AUED code construction in Section 2.2, codewords are formed by concatenation of information bits and one or more groups of check bits depending on how we want to construct code. For example, we need to decide what the length of a codeword should be. From our point of view of network construction, every code is treated in the same way except the number of networks and its applications. So we will discuss a general algorithm for network construction and demonstrate some particular cases.

#### 3.1.1 A New Algorithm

According to the method shown in Section 2.2, we know that the codeword is the form of $XR_{1,i_1}R_{2,i_2}...R_{k,i_k}$ where X represents the information bits and $R_{j,i_j}, 1 \le j \le k$, is the check bits. As we presented the method of t-EC/d-Ed/AUED code construction in Section 2, we need to find $R_{j,i_j}$, which is the row $i_j$ of the matrix $M_j$. We assume that matrix $M_j$ has been already known after computing the cardinality of $A_j$ and $\Delta$ values. $R_{j,i_j}$, can be calculated in two different ways depending on asymmetric distance *A* value which may be 1 or greater than 1. In each case, the row $i_j$ has to be computed by using either the formula

$$i_j = \left\lfloor L(X)/(d-t+1+2.\sum_{f=1}^{j-1}\Delta f) \right\rfloor$$

or $i_j = \lfloor L(X)/2^{a_j} \rfloor$, where the value of $a_j$ satisfies the relation $2^{a_j} \le d-t+2.\sum_{f=1}^{j-1}\Delta f < 2^{a_j+1}$ with respect to the value of $\Delta$.

Here we will present a general algorithm of code construction in the framework of neural computing. Before we describe the algorithm, the following notations will be used in the following algorithm.

$k$ = *length* of the information bits $X$

$n$ = number of columns in the matrix $M_j$

$m$ = number of rows in the matrix $M_j$

$Z = d-t+1+2.\sum_{f=1}^{j-1}\Delta f$ *for* $\Delta > 1$ *and* $= 2^{a_j}$ *for* $\Delta = 1$.

The network is shown in Figure 2. There are $k$ inputs to the network, which consists of associative memories comprising of two layers of neurons. In the hidden layer (also referred to as layer 1), there are $m$ neurons which represent the number of rows in the matrix $M_j$ which is assumed to be given. In layer 1, neurons are named $h_1, h_2, ..., h_m$, going from left to right. Every input is connected with each neuron of layer 1. Layer 2 has $n$ neurons which will produce the value of $R_{j,i_j}$. Similarly, neurons at layer 2 have names $z_1, z_2, ..., z_n$, going from left to right. Every neuron of layer I is connected to every neuron of layer 2.

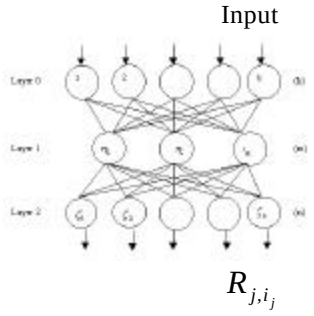



Figure 2 Network for Code Construction

We denote the weight $w_{ij}^{01}$ of the connection of the $ith$ neuron of the input I with the $jth$ neuron of layer 1. $w_{ij}^{01} = 1, 1 \le i \le m \; and \; 1 \le j \le n$.

We use $w_{ij}^{01}$ to denote the weight of the connection of the *ith* neuron of layer I with the $jth$ neuron of layer 2. The values of $w_{ij}^{12}$ are assigned by the elements of the matrix $M_j$ in the following way. In this case, we denote that $a_{ij}$ is the element at the $ith$ row and the $jth$ column of the matrix $M_j$.

i.e. For $1 \le i \le m, 1 \le j \le n$. $w_{ij}^{12} = a_{m-i+1,j}$

For each neuron of layer 1, the hard limiter activation function [Figure 3a] is used as the activation function, while the threshold logic function[10] [Figure 3b] is used for the neurons of layer 2 [10][12].

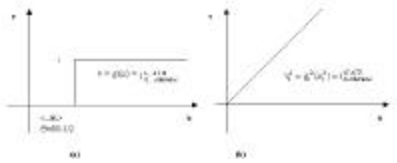


Figure 3 Activation functions used in the network

In the network, the information part of the received word is passed through the first layer and we allow the network to progress until it falls into a stable situation. In this case, the output of layer 1 determines the row number of the matrix $M_j$ and layer 2 produces the appropriate check bits for the given information part *X*.

We introduce the following variables and activation functions to show how our network performs.

(1) The initial input $v_j^0, 1 \le j \le N$

(2) The output of neuron $t$ in the layer 1 $v_t^1, 1 \le t \le M$

(3) The output of neuron $i$ in the layer 2 $v_i^2, 1 \le i \le N$

Let $g_t^1$ and $g_j^2$ be the activation functions for neurons of layer 1 and layer 2 respectively, where $1 \le t \le m$ and $1 \le j \le n$. In other words, $g_t^1$ is the activation function for neuron $h_t$, of layer 1 while $g_j^2$ is for neuron $z_j$ of layer 2. $g_t^1$ is a hard limiter activation function on the weighted sum of given inputs $v_j^0$, where $1 \le j \le n$.

Let $u_t^1 = \sum_j^k w_{jt}^{01} v_j^0, 1 \le t \le m \text{ and } v_t^1 = g_t^1(u_t^1),$

$$i.e. \quad v_t^1 = g_t^1(u_t^1) = \begin{cases} 1, \text{ if } S = m-t+1 \\ 0, \text{ otherwise} \end{cases} \qquad (1)$$

where $S = \lfloor (k - u_t^1)/Z \rfloor$.

The output values of the neurons of layer 2 are determined by the threshold logic function $g_i^2$ [10][12].

Let $u_i^2 = \sum_j^m w_{ji}^{12} v_j^1, 1 \le i \le n$ then we have,

$$i.e. \quad v_i^2 = g_i^2(u_i^2) = \begin{cases} u_i^2 & \text{if } u_i^2 \ge 0 \\ 0, & \text{otherwise} \end{cases} \qquad (2)$$

In this function, the output of $g_i^2$ will be either 0 or I since the value of $u_i^2$ is 0 or 1.

### 3.2 Illustrative Example

Here we will demonstrate our network with an illustrative example. We will use an example of Code II shown in Section 2.3.1.

#### 3.2.1 Example

In this example, we present the demonstration of construction of 2-EC/5-ED/AUED code which contains codewords of the form $XR_{1,i_1}R_{2,i_2}$ where $R_{1,i_1} \in A_1$, $R_{2,i_2} \in A_2$ and with $\Delta_1 = 1$ and $\Delta_2 = 2$. A formal way for constructing this code has been illustrated in Section 2.3.2 and named as Code II. According to Section 2.3.2, we know that the length of the information bits $X$, n'= 16 , and $d = 5$, $t = 2$. Since $\Delta_1 = 1$, we compute the cardinality of $A_1, |A_1| = 8$ by using the formula $|A| = 2^{\lceil \log(n'+1) \rceil - a_1}$, where $a_1$ is such that $2^{a_1} \le d - t + 1 < 2^{a_1+1}$, $i.e., a_1 = 2$.

$$M_1 = \begin{bmatrix} 0\,0\,0 \\ 0\,0\,1 \\ 0\,1\,0 \\ 0\,1\,0 \\ 0\,1\,1 \\ 1\,0\,0 \\ 1\,0\,1 \\ 1\,1\,0 \\ 1\,1\,1 \end{bmatrix}$$

So the binary representation of the numbers $0,1,2,...,2^{\lceil \log(n'+1) \rceil - a_1} - 1$ are the rows in the $|A_1| \times r_1$, matrix $M_1$. In this case, $r_1 = 3$ which is computed according to Section 2.2. Thus the matrix $M_1$, will be

Similarly, according to Section 2.2 and Section 2.3.2, we have $|A_2| = 3$ by the formula $|A| \ge \left\lceil (n'+1)/(d - t + 1 + 2.\sum_{f=1}^{j-1} \Delta f) \right\rceil$. So the $|A_2| \times r_2$ matrix $M_2$ will be

$$M_2 = \begin{bmatrix} 0\,0\,0\,0 \\ 0\,0\,1\,1 \\ 1\,1\,1\,1 \end{bmatrix}$$

From the above facts, we will construct a network for finding two groups of check bits $R_{1,i_1}$ and $R_{2,i_2}$. *Let* the information bits $X$ of the received word be 100 000000 1111111. We initially find the appropriate check bits, $R_{1,i_1}$ for $X$. In this case, since the number of rows in the matrix $M_1$ is 8 and the number of columns is 3, we have $m = 8$ and $n = 3$. Also the length of the information bits, k is 16. Since $\Delta_1 = 1$, then $Z = 2^{a_1} = 2^2 = 4$.

The weights of the connection between input layer 0 and layer 1 are given by

$w_{ij}^{01} = 1, \text{ where } i = 1,2,...16 \text{ and } j = 1,2,...8.$

Inputs for the layer 0, i.e. bits of the word received, are

$v_1^0 = 1, v_2^0 = 0, v_3^0 = 0, v_4^0 = 0, v_5^0 = 0, v_6^0 = 0, v_7^0 = 0, v_8^0 = 0,$

$v_9^0 = 0, v_{10}^0 = 1, v_{11}^0 = 1, v_{12}^0 = 1, v_{13}^0 = 1, v_{14}^0 = 1, v_{15}^0 = 1, v_{16}^0 = 1$

According to the proposed network, we need to find the weighted sum of these inputs. Since the weight of the each connection between layer 0 and layer 1 is 1, the weighted sum of these inputs will be the same. Thus

$$u_t^1 = \sum_j^n w_{jt}^{01} v_j^0 = 1.1 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.1 + 1.1 + 1.1 + 1.1 + 1.1 + 1.1 + 1.1 = 8$$

We get $Z = 4$ and $S = \lfloor (k - u_t^1)/Z \rfloor$, $S = \lfloor (16-8)/4 \rfloor = 2$.

We will compute the outputs of neurons in the layer 1 using the equation (1) as follows:

$v_1^1 = g_1^1(u_1^1) = 0, v_2^1 = g_2^1(u_2^1) = 0, v_3^1 = g_3^1(u_3^1) = 0, v_4^1 = g_4^1(u_4^1) = 0$

$v_5^1 = g_5^1(u_5^1) = 0, v_6^1 = g_6^1(u_6^1) = 1, v_7^1 = g_7^1(u_7^1) = 0, v_8^1 = g_8^1(u_8^1) = 0,$

The weights of synapse connecting layer 1 and layer 2 are :

$w_{11}^{12} = 1, w_{21}^{12} = 1, w_{31}^{12} = 1, w_{41}^{12} = 1, w_{51}^{12} = 0, w_{61}^{12} = 0, w_{71}^{12} = 0, w_{81}^{12} = 0$

$w_{12}^{12} = 1, w_{22}^{12} = 1, w_{32}^{12} = 0, w_{42}^{12} = 0, w_{52}^{12} = 1, w_{62}^{12} = 1, w_{72}^{12} = 0, w_{82}^{12} = 0$

$w_{13}^{12} = 1, w_{23}^{12} = 0, w_{33}^{12} = 1, w_{43}^{12} = 0, w_{53}^{12} = 1, w_{63}^{12} = 0, w_{73}^{12} = 1, w_{83}^{12} = 0$

Inputs for neurons at layer 2 are :

$v_1^1 = 0, v_2^1 = 0, v_3^1 = 0, v_4^1 = 0, v_5^1 = 0, v_6^1 = 1, v_7^1 = 0, v_8^1 = 0,$

The weighted sum of these inputs are:

$u_1^2 = \sum_j^m w_{j1}^{12} v_j^1 = 1.0 + 1.0 + 1.0 + 1.0 + 0.0 + 0.1 + 0.0 + 0.0 = 0$

$u_2^2 = \sum_j^m w_{j2}^{12} v_j^1 = 1.0 + 1.0 + 0.0 + 0.0 + 1.0 + 1.1 + 0.0 + 0.0 = 1$

$u_3^2 = \sum_j^m w_{j3}^{12} v_j^1 = 1.0 + 0.0 + 1.0 + 0.0 + 1.0 + 0.1 + 1.0 + 0.0 = 0$

Since we have defined the threshold logic function as

an activation function for each neuron, the outputs of neurons in the layer 2 are :

$$v_1^2 = g_2^2(u_1^2) = 0$$
$$v_2^2 = g_2^2(u_2^2) = 1$$
$$v_3^2 = g_3^2(u_3^2) = 0$$

Thus for the given information bits $X$ ; 1000000001111111 the check bits $R_{1,j_1}$, are 010.

We can also find the check bits $R_{2,j_2}$ by using matrix $M_2$. In this case, since the number of rows in the matrix $M_2$ is 3 and the number of columns is 4, we have $m = 3$ and $n = 4$. Also the length of the information bits, $k$ is 16. Since $\Delta_1 = 2 > 1$, then $Z = d - t + 1 + 2.\Delta_1 = 5 - 2 + 1 + 2 = 6.$

The weights of the connections between input layer 0 and layer 1 are given by

$$w_{ij}^{01} = 1, where\ i = 1,2,...,16\ and\ j = 1,2,...,3.$$

Inputs for the layer 0, i.e. bits of the word received, are

$$v_1^0 = 1, v_2^0 = 0, v_3^0 = 0, v_4^0 = 0, v_5^0 = 0, v_6^0 = 0, v_7^0 = 0, v_8^0 = 0,$$
$$v_9^0 = 0, v_{10}^0 = 1, v_{11}^0 = 1, v_{12}^0 = 1, v_{13}^0 = 1, v_{14}^0 = 1, v_{15}^0 = 1, v_{16}^0 = 1$$

According to the network, we need to find the weighted sum of these inputs. Since the weight of the each connection between layer 0 and layer 1 is 1, the weighted sum of these inputs will be the same.
Thus

$$u_t^1 = \sum_j^m w_{jt}^{01} v_j^0 = 1.1 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 + 1.0 +$$
$$1.1 + 1.1 + 1.1 + 1.1 + 1.1 + 1.1 + 1.1 + 1.1 = 8$$

We get Z = 4 and $S = \lfloor (k - u_t^1)/Z \rfloor S \lfloor (16-8)/6 \rfloor = 1$, then we will compute the outputs of neurons in the layer 1 using the equation (1) as follows: follow:

$$v_1^1 = g_1^1(u_1^1) = 0$$
$$v_2^1 = g_2^1(u_2^1) = 1$$
$$v_3^1 = g_3^1(u_3^1) = 0$$

The weights of synapse connecting layer 1 and layer 2 are

$$w_{11}^{12} = 1, w_{21}^{21} = 0, w_{31}^{12} = 0$$
$$w_{12}^{12} = 1, w_{22}^{21} = 0, w_{32}^{12} = 0$$
$$w_{13}^{12} = 1, w_{23}^{21} = 1, w_{33}^{12} = 0$$
$$w_{14}^{12} = 1, w_{24}^{21} = 1, w_{34}^{12} = 0$$

Inputs for neurons at layer 2 are :

$$v_1^1 = 0, v_2^1 = 1, v_3^1 = 0$$

The weighted sum of these inputs are:

$$u_1^2 = \sum_j^m w_{j1}^{12} v_j^1 = 1.0 + 0.1 + 0.0 = 0$$
$$u_2^2 = \sum_j^m w_{j2}^{12} v_j^1 = 1.0 + 0.1 + 0.0 = 0$$
$$u_3^2 = \sum_j^m w_{j3}^{12} v_j^1 = 1.0 + 1.1 + 0.0 = 1$$
$$u_4^2 = \sum_j^m w_{j4}^{12} v_j^1 = 1.0 + 1.1 + 0.0 = 1$$

Since we have defined the threshold logic function (2) as an activation function for
each neuron, the outputs of neurons in the layer 2 are :

$$v_1^2 = g_2^2(u_1^2) = 0$$
$$v_2^2 = g_2^2(u_2^2) = 0$$
$$v_3^2 = g_2^2(u_3^2) = 1$$
$$v_4^2 = g_2^2(u_4^2) = 1$$

Thus for the given information bits
X : 1000000001111111, we get the check bits $R_{2,i_2}$ ,0011. Thus the required codeword is

100 0000001111111 010 0011
$X$ $R_{1,i_1}$ $R_{2,i_2}$

which can be checked using Table 3. 2-EC/5-ED/AUED Code II.

In this way, we can construct the required t-EC/d-ED/AUED code which contains codewords of the form $XR_{1,i_1}R_{2,i_2}$.

### 3.3 A Scheme for Error Detection/Correction

We have described error correcting network for linear codes in [1]. According to the Section 2.4 and Section 2.4.1, detecting and correcting algorithm is almost the same as the algorithm in Section 3.3. So we will use the same network structure of Section 3.3. In this case we will show detecting and correcting a single error. The remaining part will be left for future research. We will demonstrate the network using the t-EC/d-Ed/AUED codes in the next section.

#### 3.3.1 Example

In this section, we will demonstrate the neural network approach error detecting and correction. As we mentioned in Section 2.2, we must have a systematic t-EC/d-ED parity check code F which has a parity check matrix H shown in 2.4.1.
Let the word 110 0000001111111 00011000 be the information bits of the received word. We assume that these received words contains some error. In this problem, since the number of row in the check matrix is 13 and the length of the codeword is 16, we have M = 13 and *N = 16*. According to Section 3.3, the weights of the synapse connecting between input layer 0 and layer 1 are defined as
$w_{ij}^{01} = h_{ji}, 1 \le i \le N\ and\ 1 \le j \le M,$ where $w_{ij}^{01}$ is the weight of the *ith* neuron of the input layer with the *jth* neuron of layer 1, and $h_{ij}$ is an element of row *ith* and column *jth* of matrix $H$.
Inputs for the layer 1, i.e. bits of the word received, are

$$v_1^0 = 1, v_2^0 = 1, v_3^0 = 0, v_4^0 = 0, v_5^0 = 0, v_6^0 = 0, v_7^0 = 0, v_8^0 = 0,$$
$$v_9^0 = 0, v_{10}^0 = 1, v_{11}^0 = 1, v_{12}^0 = 1, v_{13}^0 = 1, v_{14}^0 = 1, v_{15}^0 = 1, v_{16}^0 = 1$$

According to the proposed network, we need to find the weighted sum of these inputs
as follows:

$$u_1^1=\sum_j^N w_{j1}^{01}v_j^0=0, u_2^1=\sum_j^N w_{j2}^{01}v_j^0=0, u_3^1=\sum_j^N w_{j3}^{01}v_j^0=0, u_4^1=\sum_j^N w_{j4}^{01}v_j^0=1$$

$$u_5^1=\sum_j^N w_{j5}^{01}v_j^0=1, u_6^1=\sum_j^N w_{j6}^{01}v_j^0=1, u_7^1=\sum_j^N w_{j7}^{01}v_j^0=3, u_8^1=\sum_j^N w_{j8}^{01}v_j^0=3$$

$$u_9^1=\sum_j^N w_{j9}^{01}v_j^0=3, u_{10}^1=\sum_j^N w_{j10}^{01}v_j^0=2, u_{11}^1=\sum_j^N w_{j11}^{01}v_j^0=2, u_{12}^1=\sum_j^N w_{j12}^{01}v_j^0=2$$

$$u_{13}^1=\sum_j^N w_{j13}^{01}v_j^0=3$$

After applying the activation function (1) as shown

$$i.e.\quad v_t^1=g^1(u_t^1)=\begin{cases}1 & if\ u_t^1 \mod 2=1\\ -1 & otherwise\end{cases}$$

the outputs of neurons in the layer 1 are:

$$v_1^1=g^1(u_1^1)=-1, v_2^1=g^1(u_2^1)=-1, v_3^1=g^1(u_3^1)=-1, v_4^1=g^1(u_4^1)=1$$

$$v_5^1=g^1(u_5^1)=1, v_6^1=g^1(u_6^1)=1, v_7^1=g^1(u_7^1)=1, v_8^1=g^1(u_8^1)=1$$

$$v_9^1=g^1(u_9^1)=1, v_{10}^1=g^1(u_{10}^1)=-1, v_{11}^1=g^1(u_{11}^1)=-1, v_{12}^1=g^1(u_{12}^1)=-1$$

$$v_{13}^1=g^1(u_{13}^1)=-1$$

According to Section 3.3, the weights of synapse connecting layer 1 and layer 2 are defined as follow:

For $1\le i\le M, 1\le j\le N$

$$w_{ij}^{12}=\begin{cases}1 & if\ h_{ij}=1\\ -1 & otherwise\end{cases}$$

Inputs for neurons at layer 2 are :

$$v_1^1=-1, v_2^1=-1, v_3^1=-1, v_4^1=1$$

$$v_5^1=1, v_6^1=1, v_7^1=1, v_8^1=1$$

$$v_9^1=1, v_{10}^1=-1, v_{11}^1=-1, v_{12}^1=-1$$

$$v_{13}^1=1$$

The weighted sum of these inputs are:

$$u_1^2=\sum_j^M w_{j1}^{12}v_j^1=1, u_2^2=\sum_j^M w_{j2}^{12}v_j^1=13, u_3^2=\sum_j^M w_{j3}^{12}v_j^1=1, u_4^2=\sum_j^M w_{j4}^{12}v_j^1=-3$$

$$u_5^2=\sum_j^M w_{j5}^{12}v_j^1=-3, u_6^2=\sum_j^M w_{j6}^{12}v_j^1=-3, u_7^2=\sum_j^M w_{j7}^{12}v_j^1=1, u_8^2=\sum_j^M w_{j8}^{12}v_j^1=1$$

$$u_9^2=\sum_j^M w_{j9}^{12}v_j^1=1, u_{10}^2=\sum_j^M w_{j10}^{12}v_j^1=1, u_{11}^2=\sum_j^M w_{j11}^{12}v_j^1=1, u_{12}^2=\sum_j^M w_{j12}^{12}v_j^1=1$$

$$u_{13}^2=\sum_j^M w_{j13}^{12}v_j^1=-3, u_{14}^2=\sum_j^M w_{j14}^{12}v_j^1=-3, u_{15}^2=\sum_j^M w_{j15}^{12}v_j^1=-3, u_{16}^2=\sum_j^M w_{j16}^{12}v_j^1=1$$

Here we use the activation function (2) shown

$$ie.\quad v_i^2=g^2(u_i^2)=\begin{cases}1 & if\ u_i^2\ge q\\ -1 & otherwise\end{cases}$$

and threshold Θ=M-1/2=13-1/2=12.5 and, the outputs of neurons in the layer 2 are:

$$v_1^2=g^2(u_1^2)=0, v_2^2=g^2(u_2^2)=1, v_3^2=g^2(u_3^2)=0, v_4^2=g^2(u_4^2)=0$$

$$v_5^2=g^2(u_5^2)=0, v_6^2=g^2(u_6^2)=0, v_7^2=g^2(u_7^2)=0, v_8^2=g^2(u_8^2)=0$$

$$v_9^2=g^2(u_9^2)=0, v_{10}^2=g^2(u_{10}^2)=0, v_{11}^2=g^2(u_{11}^2)=0, v_{12}^2=g^2(u_{12}^2)=0$$

$$v_{13}^2=g^2(u_{13}^2)=0, v_{14}^2=g^2(u_{14}^2)=0, v_{15}^2=g^2(u_{15}^2)=0, v_{16}^2=g^2(u_{16}^2)=0$$

After the first phase, we have detected that second position of the given word is in error. In the second phase, we use the XOR network shown in Section 3.3 with the inputs of the corresponding bit positions of the output of phase 1( 0 10 0 0 0 0 0 0 0 0 0 0 0 0 0 0)and the received word (1 10 000000 1 1 1 1 1 1 1 ).Then according to the error correcting phase shown in Section 3.3.2, the XOR network produces the correct codeword ( 10 0 0 0 0 0 0 0 1 1 1 1 1 1 1 ).

## 4. Conclusion Remarks

In the above paper, we discussed an algorithm of the construction of neural networks for *2EC/5ED/AUED Code II.*. We appreciate many researchers for their excellent research work on error detecting/correcting codes which we referred many places in this paper.

## 5. References


[1]. Norman L. Biggs "Discrete Mathematics", Revised edition 1989, Oxford University Press.

[2]. M. Blaum and H. V. Tilborg, "On t-Error Correcting/All Unidirectional Error Detecting Codes", *IEEE Transactions on Computers* Vol 38, pp-1493-1501, Nov. 1989.

[3]. B. Bose, "Theory and design of unidirectional eror codes," Ph.D. Dissertation, Dep. Computer Sci. Eng., Southern Methodist Univ., Dallas, TX. May 1980.

[4]. B. Bose, "On systematic SEC/MUED code," in *Proceeding FTCS*. Vol. 11, June 1981, pp. 265-267.

[5]. B. Bose and T. R. Rao, "Theory of Unidirectional Error Correcting/ Detecting Codes", *IEEE Transactions on Computer* Vol c-31, pp-521-530. June, 1982.

[6]. B. Bose and D. K. Pradhan, "Optimal Unidirectional Error Detecting/ Correcting Codes", *IEEE Transactions on Computers* Vol c-31, pp 564-568, June, 1982.

[7]. B. Bose, "On Unordered Codes", *IEEE Transactions on Computers* Vol 40, pp-125-131, Feb. 1991.

[8]. B. Bose and D. J. Lin, "Systematic Unidirectional Error-Detecting Codes" *IEEE Transactions on Computers* Vol 34, pp-1026-1032, Nov. 1985.

[9]. R. W. Cook, W. H. Sisson, T. F. Storey, and W. W. Toy, "Design of a self-checking microprogram control," *IEEE Transactions on Computers*, Vol. C-22, pp. 255-262, Mar. 1973.

[10]. D. J. Evans, M. Adamopoulos, S.Kortesis, and K. Tsouros "Searching sets of properties with neural network." Parallel Computing 16(1990) 279-285 North-Holland.

[11]. D. J. Lin and B. Bose, "Theory and design of t-error correcting and d(d>t)-unidirectional error detecting(t-EC d-UED) codes," *IEEE Transactions on Computers*, Vol 37, pp. 433-439, Apr. 1988.

[12]. R. P. Lippmann, "Introduction to computing with neural nets." *IEEE Transactions on Acoustics, Speech and Signal Processing* (April 1987) pp-4-22.

[13]. D. Nikolos, N. Gaitanis, and G. Philokyprou, "Systematic t-Error Correcting/All Unidirectional Error Detecting Codes", *IEEE Transactions Computer* Vol c-35, pp-394-402, May, 1986.

[14]. D. Nikolos, "Theory and Design of t-Error Correcting/ d-Error Detecting ( d > t )and All Unidirectional Error Detecting Codes", *IEEE Transactions on Computers* Vol 40, pp-132-142, Feb. 1991.

[15]. B. Parhami and A. Avizienis, " Detection of storage errors in mass memories using low-cost arithmetic codes," *IEEE Transactions on Computers*, Vol C-27, pp 302-308, Apr. 1978.

[16]. D.K. Pradhan and S.M. Reddy, "Fault-tolerant failsafe logic networks," in *Proceeding IEEE COMPCON*, San Francisco, CA, Mar. 1977, p.363.

[17]. D. K. Pradhan, "A new Class of Error-Correcting/Detecting Codes for Fault-Tolerant Computer Applications", *IEEE Transactions on Computers* Vol c-29, pp.471-481. June, 1980.

[18]. D. L. Tao, C. R. P. Hartmann, and P.K. Lala, "An efficient class of unidirectional error detecting/correcting codes," *IEEE Transactions, Computer*, Vol. 37, pp 879-882, July 1988.

[19]. D. Tasar and V. Tasar, " A study of intermittent faults in digital computers", in FIProc. AFIPS Conference, 1977, pp 807-811.

[20]. J. F. Wakerly, "Detection of unidirectional multiple errors using low cost arithmetic codes," *IEEE Transactions on Computers*," Vol C-24, pp 210-212, Feb. 1975.

[21]. J.H. Weber, C. de Vroedt, and Dick E. Boekee, "Bounds and Constructions for Binary Codes of Length Less Than 24 and Asymmetric Distance Less Than 6", IEEE Transactions on Information Theory, Vol. 34, No. 5. Sept. 1988.

[22]. ---, "systematic t-error correcting/all unidirectional detecting codes," *IEEE Transactions on Computers*, Vol C-35, pp 394-402, May 1986.

[23]. Maung Maung Htay, S. Sitharama Iyengar, and Si-Qing Zheng, "Correcting Errors in Neural Network" IEEE 27th Southeastern Symposium on System Theory, Proceeding, pp 386-391, March 12-14, 1995.

[24]. Maung Maung Htay, S. Sitharama Iyengar, and Si-Qing Zheng, " Systematic Unidirectional Error-Detecting Codes with Neural Network", ISTEAD 10th Parallel and Distributed Computing and Systems, Proceeding pp 95-100, October 28-31, 1998.

[25]. Maung Maung Htay, S. Sitharama Iyengar, and Si-Qing Zheng, "t-Error Correcting/d-Error Detecting (d>t) and All Unidirectional Error Detecting Codes with Neural Network(Part I), ITCC, 2001, IEEE, Las Vegas Proceeding 529-536 April 2-4, 2001.