Using Hamming Network to Decoding Binary Cyclic Code

Hind Abd Al-Razzaq*

Received on:1/11/2009
Accepted on:2/6/2011

Abstract

This work, efforts are concentrated on solving the problem of decoding binary cyclic code, using hamming neural network. Therefore, this work shows the ability of hamming network in solving one of the important problems in coding theory. It presents the results of applying hamming network as a decoding algorithm for cyclic code. The results prove the relative efficiency of hamming network in decoding large linear cyclic codes compared with other decoding algorithms.

Keywords: Hamming Network, Neural Network, Cyclic Code, Block Code

استخدام شبكة الهامنك في تحليل الشفرة الدورية الثنائية

الخلاصة

تركز الاهتمام على مشكلة تحليل الشفرات الدورية الثنائية من خلال استخدام شبكه الهامنك لخلايا الأعصاب. يوضح البحث إمكانية الشبكة في حل واحدة من المشاكل كبيرة الأهمية ضمن نظرية الترميز. تم استخدام نتائج هذا البحث لجعل شبكة الهامنك المعتمدة في هذا العمل كخوارزمية تحليلية للشفرات الدورية. برهنت النتائج أيضا قدرتها على تحليل الشفرات الدوريةالكبيرة مقارنة مع خوارزميات تحليلية أخرى.

1-Introduction

In recent years, the demand for efficient and reliable digital data transmission systems has been accelerated by the increasing use of automatic data processors and the rising need for long range communications. One of the serious problems in any high-speed data transmission system is the occurrence of errors. To control these errors, there are three techniques in use [1,2,3].

1-Echo checking

2-Automatic repeat request (ARQ)

3-Forward error correction (FEC)

2-Types of Errors

On memory less channels, the noise affects each transmitted symbol independently.

Hence transmission errors occur randomly in the received sequence, and memory less channels are called (Random Error Channels), such as satellite channel. The codes devised for correcting such errors called **Random error correcting codes**.

On channels with memory, noise is independent from transmission to another, hence errors occur in clusters or bursts and called Burst Error channels, such as radio channel. The codes devised for correcting such errors

***Building and Construction Engineering Department, University of Technology / Baghdad**

https://doi.org/10.30684/etj.29.11.1
2412-0758/University of Technology-Iraq, Baghdad, Iraq

called ***Burst error correcting codes*** [2].

3-Linear block codes

• An (*n*,*k*) linear block code is a k-dimensional subspace of a finite field F^n .

Sums, differences, and scalar multiples of codewords are also codewords.

• A group code over an additive group G is closed under sum and difference.

• An (*n*,*k*) LBC over F = GF(q) has $M = q^k$ codewords and rate k/n.

• A linear block code C can be defined by two matrices.

- Generator matrix G: rows of G are basis for C, i.e., C = {mG : m Є F^k }
- Parity-check matrix H span $C^{\perp}$, hence C = {c Є F^n : cH^T = 0}

• The Hamming weight of an *n*-tuple is the number of nonzero components.

• The minimum weight w^* of a block code is the Hamming weight of the nonzero codeword of minimum weight.

• The minimum distance of every LBC equals the minimum weight: $d^* = w^*$.

• The minimum weight of a linear block code is the smallest number of linearly dependent columns of any parity-check matrix [4,5].

4-Bounds on minimum distance

The minimum distance of a block code is a conservative measure of the quality of an error control code.

• A large minimum distance guarantees reliability against random errors.

• However, a code with small minimum distance may be reliable—provided the probability of sending codewords with nearby codewords is small.

We use minimum distance as the measure of a code's reliability because:

• A single number is easier to understand than a weight/distance distribution.

• The guaranteed error detection and correction ability are

- detection: $e = d^* - 1$ (4.1)
- correction: $t <= [(d^* - 1)$ (4.2)

• Algebraic codes covered in the course are limited by minimum distance—these codes cannot correct more than t errors even if there is only one closest codeword [3,5].

5-Binary Cyclic Codes

Cyclic codes are a subset of the class of linear block codes which satisfy the following Description:

- If the components of an *n*-tuple $v=(v_0,v_1,\ldots,v_{n-1})$ are cyclically shifted *i* places to the right , the resultant *n*-tuple would be $v^{(i)}=(v_{n-i},v_{n-i+1},\ldots,v_{n-1} , v_0,v_1,\ldots,v_{n-i-1})$. (5.1)
- Cyclically shifting *v i* places to the right is equivalent to cyclically shifting *v n-i* places to the left.
- An (*n*,*k*) linear code ***C*** is called a cyclic code if every cyclic shift of a code vector in ***C*** is also a code vector in ***C*** [1,6].

6-SyndromComputation

- Let r= ($r_0,r_1,\ldots,r_{n-1}$) be the received vector. The *syndrome* is calculated as

$$\boldsymbol{s = r.\ H^T} \qquad (6.1)$$

, where **H** is the parity-check matrix.

- If syndrome is not identical to zero, r is not a code vector and the presence of errors has been detected.
- Dividing $r(x)$ by the generator. Polynomial $g(x)$, we obtain

$$\boldsymbol{r(x)= a(x)g(x) + s(x)} \qquad (6.2)$$

- The n-k coefficients of $\boldsymbol{s(x)}$ form the syndrome s. we call $\boldsymbol{s(x)}$ the syndrome.
- If C is a systematic code, then the syndrome is simply the vector sum of the received parity digits and the parity-check digits recomputed from the received information digits.

Let $\boldsymbol{s(x)}$ be the syndrome of a received polynomial $\boldsymbol{r(x)}$. Then the remainder $\boldsymbol{s^{(1)}(x)}$ resulting from dividing $x\ \boldsymbol{s(x)}$ by the generator polynomial $\boldsymbol{g(x)}$ is the syndrome of $\boldsymbol{r^{(1)}(x)}$, which is a cyclic shift of $\boldsymbol{r(x)}$[6]. **7-The Hamming Network**

The Hamming network is a straightforward associative memory. It calculates the Hamming distance between the input pattern and each memory pattern, and selects the memory with the smallest Hamming distance. The network output is the index of a prototype pattern and thus the network can be used as a pattern classifier. The Hamming network is used as the classical Hamming decoder or Hamming associative memory. It provides the minimum-Hamming-distance solution.

The Hamming network has a *J-N-N* layered architecture, as illustrated in Fig. 2. The third layer is called the memory layer, each of whose neurons corresponds to a prototype pattern. The input and hidden layers are feed forward, fully connected, while each hidden node has a feed forward connection to its corresponding node in the memory layer .Neurons in the memory layer are fully interconnected, and form a competitive sub network known as the *MAXNET*. The MAXNET responds to an input pattern by generating a winner neuron through iterative competitions. The Hamming network is implicitly recurrent due to the interconnections in the memory layer [7].

Hamming Network Algorithm [7,8]

Step1. Assign Connection Weights and Offsets

In the lower subnet:

$$W_{ij}=X^j_i/2,\theta_j=N/2, \qquad (7.1)$$

$$0 \le i \le N-1, \quad 0 \le j \le M-1$$

In the upper subnet:

$$t_{kl} = \begin{cases} 1, & k = l \\ -\epsilon, & k \ne l, \epsilon<1/M, \end{cases} \qquad (7.2)$$

$$0 \le k, l \le M-1$$

In these equations w_{ij} is the connection weight from input i to node j in the lower subnet and θ is the threshold in that node. The connection weight from node k to node l in the upper subnet is t_{kl} and all thresholds in this subnet are zero. X^j_i is element i of exemplar j.

Step2. Initialize with Unknown Input Pattern

$$\mu_j(0)=f_t(\sum_{i=0}^{N-1} w_{ij}x_i-\theta_j) \quad (7.3)$$

$$0 \le j \le M-1$$

In this equation $\mu_j(t)$ is the output of node j in the upper subnet at time t, x_i is element i of the input ,and f_t is the threshold logic nonlinearity. Here and below it is assumed that the maximum input to this nonlinearity never causes the output to saturate.

Step3.Iterate until Convergence

$$\mu(t+1)=f_t(\Sigma\mu(t)-\epsilon\sum_{k\neq j}\mu_k(t)) \quad (7.4)$$

$$0 \le j,k \le M-1$$

This process is repeated until convergence after which the output of only one node remains positive.

Step4.Repeat by Going to Step2

8-System Overview

This system, which is presented in this work for decoding binary cyclic codes, is based on hamming network as mentioned earlier. Thus, the main components of this system, which we call Hamming Network for Decoding Binary Cyclic Code (HN_DBCC), are shown in figure (3).

8.1 Code Specifications

The code specifications involve the information needed by the HN_DBCC system. In this work, the description of system needs to be entered:

- length of the code(n).
- length of the information bits(k).
- error correcting capability(t).
- parity check matrix for this code(H).

9-Hamming Network algorithm to Decoding Cyclic Code

Step1: Define the list of the codeword accepted in the code:

Example: code6 (cyclic code (15, 1, 7))

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 } Codeword

Step2: Save the accepted codeword in exemplar array.

Exemplar(0) -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1

Exemplar(1) 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Step3: Assign connection weight & offset in the lower subnet this mean the Learning stage *by computing the equation* (7.1)

$\theta_j=N/2=7.5$ where N=15

$W_{ij} = X^j_i /2$ where $0 \le i \le 15$, $0 \le j \le 2$

Weight(0) -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5 -0.5

Weight(1) 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 .0.5 0.5

Step4: Assign connection weight & offset in the upper subnet by computing the equation (7.2)

Step5: Initialize with Unknown received code this mean the Testing stage ***by Compute the*** equation(7.3) Examples:

1-Unknown received code with no errors

Ex1: 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

2- Unknown received code with 3 burst error

Ex2: 1 1 1 1 1 0 0 0 1 1 1 1 1 1 1

erro

3- Unknown received code with 3 random error

Ex3: 1 1 0 1 1 1 0 1 1 1 1 1 1 1 0

error1 error2 error3

Step6: *Compute the equation* (7.4)

Step7: Iterate step6 until correct the received

Step8: when the received code correct display the message corrects and no of Exemplar match with that code define in the step1.

Show the examples in the step4 and show the output in this bellow:

Ex1: The unknown received code match the exemplar (1) in the Iteration (1)

Ex2: The unknown received code match the exemplar (1) in the Iteration (32)

Ex3: The unknown received code match the exemplar (1) in the Iteration (156)

10-Experimental work

1- Use the cyclic in this work as shown in table (1)

2- The experimental results of using traditional method to decoding as shown in table(2),figure(4)

3- The experimental results of using hamming net to decoding as shown in table(3),figure(5)

11-Results

1- Relative efficiency of hamming network in decoding large linear cyclic codes compared with other decoding

2- The hamming network needs less storage space to implement. It is only require to storing some codeword

3- When the numbers of errors is less than (t), then the hamming net is relatively efficient to correct them.

4- Hamming net efficient to correct burst errors than random errors.

5- Some times when input the unknown received code this code probabilit match with more than one codeword.

References

[1] Wassan S. Awad, " The Determination of Minimum Weight for Cyclic Codes ", M.SC. Thesis Submitted to the Dep. of Computer Science, Univ. of Technology, Baghdad- Iraq 1994.

[2] Fred Halshall, "Data Communications, Computer Networks and Open Systems", Addison-Wesley Publishers Ltd , Addition-Wesley

Puplishing Company Inc., 1996.

[3] Emanuele Betti , Emmanuela Orsini , " An introduction to Cyclic Codes ", Department of Mathematics, University of Florence, Italy, Department of Mathematics, University of Milan ,Italy , may 1,2006

[4] Samantha R. summerson, " Linear Block Codes", 30 November, 2009

[5] John Gill, " Linear block codes and group codes ", Stanford University, October8 ,2009.

[6] yughsiang S. Han , "Cyclic Codes", Graduate institute of Communication Engineering , National Taipei University Taiwan,2006.

[7] K. –L. Du and M. N. S Swamy, "Neural Networks in a Soft Computing Framework" , Springer- verlag London Limited 2006.

[8] Werner Kinnebrock "Neural Networks fundamentals, Applications, examples" Suneel Galgotia , 1995

Table (1) Code Developed in the Experimental Work

Code No.	n	k	t	Generator matrix	Parity check matrix
Code 3	15	3	2	$G(x)=1+x^3+x^6+x^9+x^{12}$	$H(x)=1+x^3$
Code 4	21	3	3	$G(x)=1+x^3+x^6+x^9+x^{12}+x^{15}+x^{18}$	$H(x)=1+x^3$
Code 5	15	2	4	$G(x)=1+x+x^3+x^4+x^6+x^7+x^9+x^{10}+x^{12}+x^{13}$	$H(x)=1+x+x^2$
Code 6	15	1	7	$G(x)=1+x+x^2+x^3+x^4+x^5+x^6+x^7+x^8+x^9+x^{10}+x^{11}+x^{12}+x^{13}+x^{14}$	$H(x)=1+x$
Code 7	21	1	10	$G(x)=1+x+x^2+x^3+x^4+x^5+x^6+x^7+x^8+x^9+x^{10}+x^{11}+x^{12}+x^{13}+x^{14}+x^{15}+x^{16}+x^{17}+x^{18}+x^{19}+x^{20}$	$H(x)=1+x$
Code 8	24	1	11	$G(x)=1+x+x^2+x^3+x^4+x^5+x^6+x^7+x^8+x^9+x^{10}+x^{11}+x^{12}+x^{13}+x^{14}+x^{15}+x^{16}+x^{17}+x^{18}+x^{19}+x^{20}+x^{21}+x^{22}+x^{23}$	$H(x)=1+x$

Table (2) Total Search Space in Traditional Method

Codes	N	T	Total
Code1	15	2	120
Code2	21	3	1561
Code3	15	4	1940
Code4	15	7	16383
Code5	21	10	1048575
Sum			1068592
Average			152656

Table (3) Total Search Space in Hamming Network

Codes	N	T	Iteration	Total
Code1	15	2	13.6	108.8
Code2	21	3	66.5	465.5
Code3	15	4	57.473	229.892
Code4	15	7	180.538	361.076
Code5	21	10	5605	5605
Sum			5923.111	6770.268

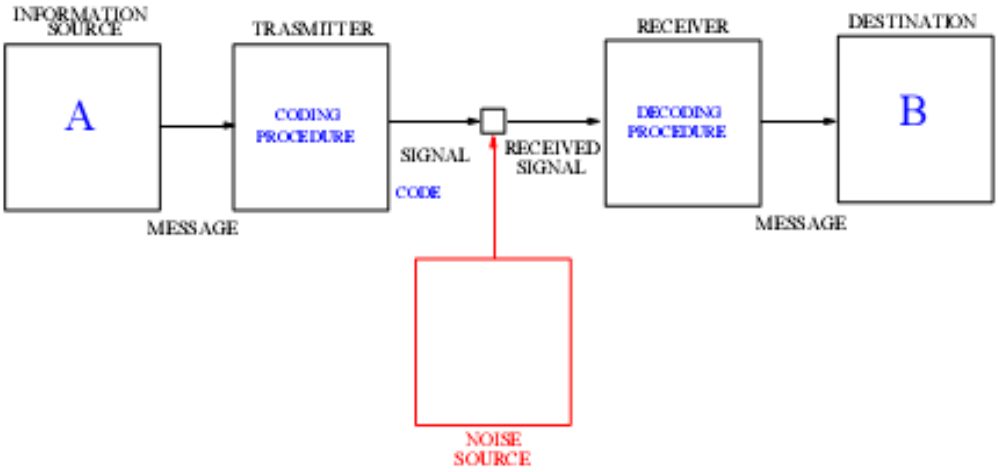


Figure (1) A communication schema

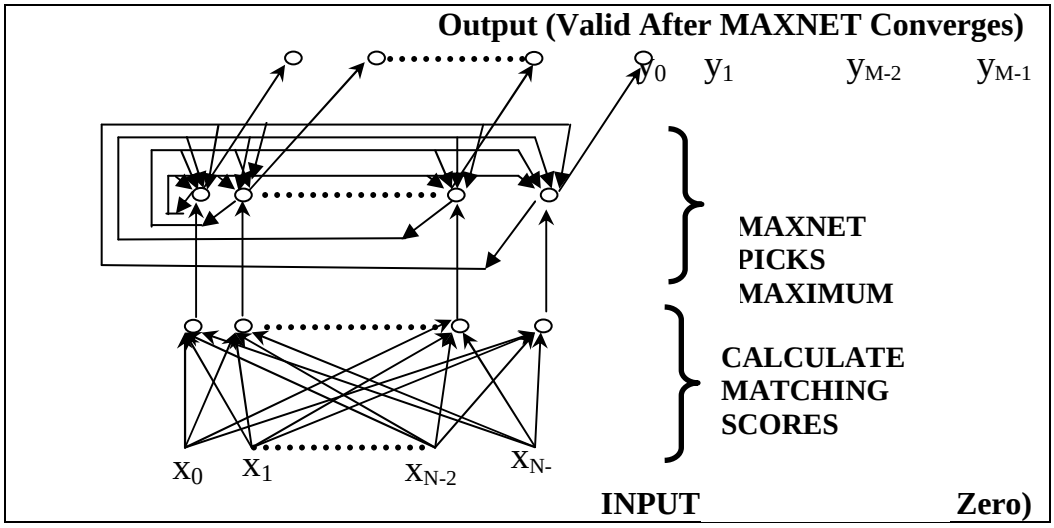
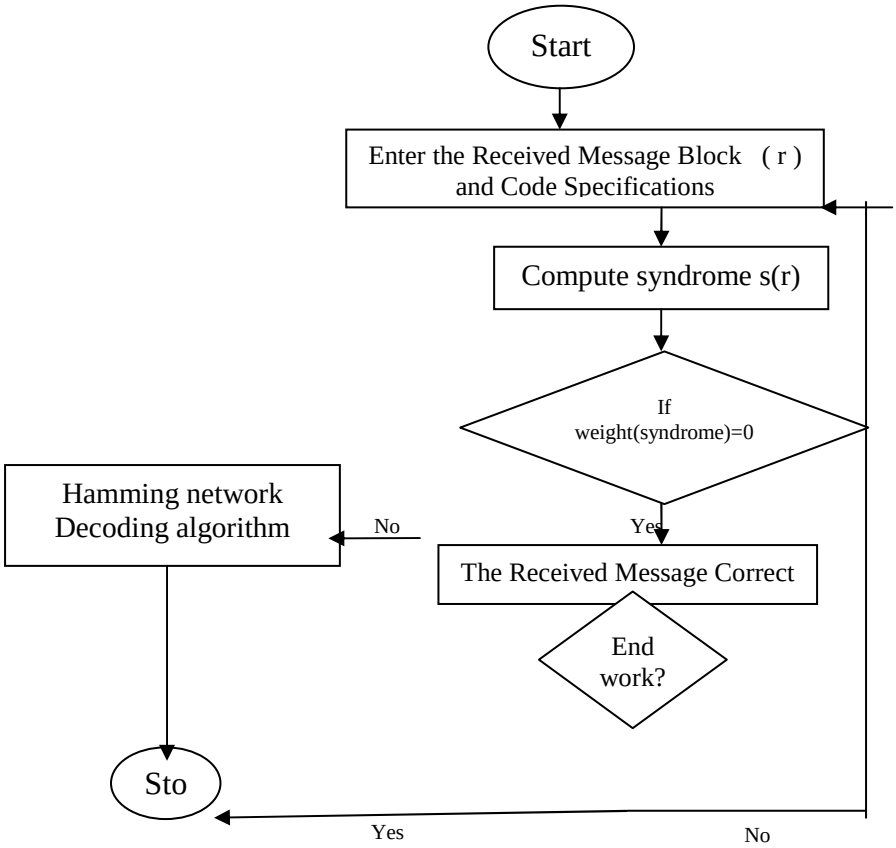


Figure (2) [7,8] A feed-forward Hamming net maximum likelihood classifier for binary inputs corrupted by noise. The lower subnet calculates *N* minus the Hamming distance to *M* exemplar patterns. The upper net selects that node with the maximum output. All nodes use threshold-logic nonlinearities never saturate.



Figure(3) The flowchart of HN_DBCC

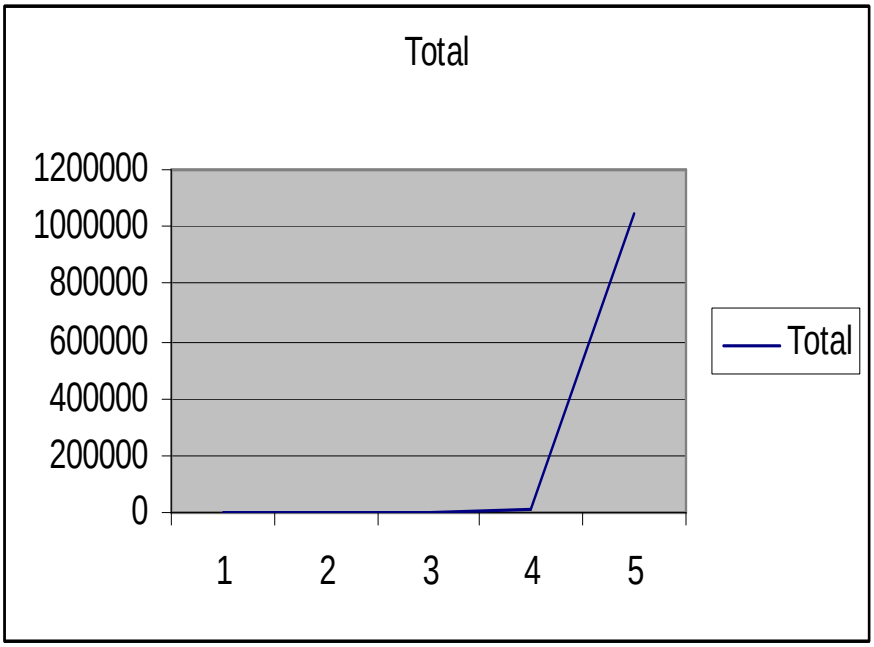


Figure (4) Total Search Space in Traditional Method

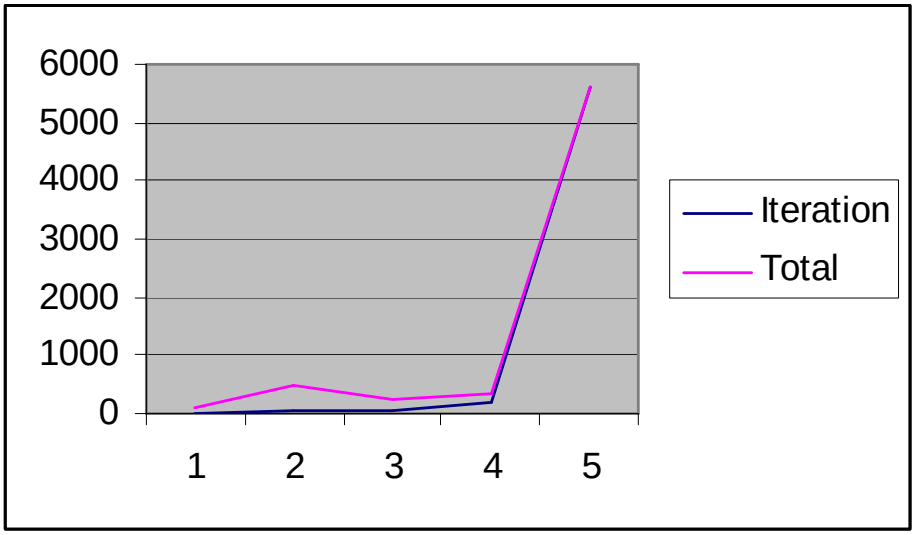


Figure (5) Total Search Space in Hamming Net