Modifying Hebbian Network for Text Cipher

Noor Adnan Ibraheem*

Received 11, November, 2009
Accepted 27, June, 2010

Abstract:

The objective of this work is to design and implement a cryptography system that enables the sender to send message through any channel (even if this channel is insecure) and the receiver to decrypt the received message without allowing any intruder to break the system and extracting the secret information.

This work modernize the feedforward neural network, so the secret message will be encrypted by unsupervised neural network method to get the cipher text that can be decrypted using the same network to get the original text.

The security of any cipher system depends on the security of the related keys (that are used by the encryption and the decryption processes) and their corresponding lengths.

In this work, the key is the final weights that are obtained from the learning process within the neural network stage, So the work can be represented as an update or development for using the neural network to enhance the security of text.

As a result for a powerful design, the resulted cipher system provides a high degree of security which satisfies the data confidentially which is the main goal of the most cryptography systems.

Key words : Hebbian Network, Neural Network, Text Security.

Introduction:

Cryptography is used to protect information to which illegal access is possible. Thus it can be applied to protect communication channels and physical databases [1].

Usually to describe any powerful cipher system it must presented from these two points of view:

- The sender point of view (this is called *encryption* process)
- The receiver point of view (this is called the *decryption* process)

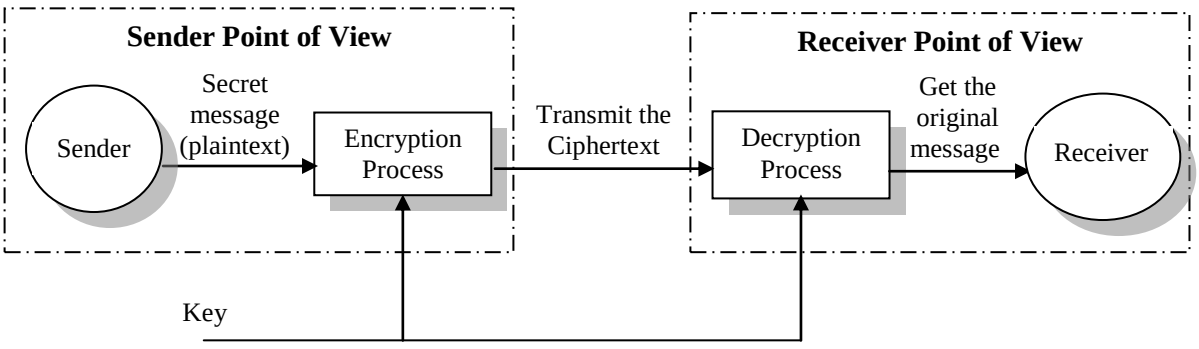

Fig 1: A Framework of a cipher system

*University of Baghdad/ College of Science for Women/ Department of Computer Science

One of the Cryptographic techniques is symmetric-key technique, which means that the same key is used for both encryption and decryption [2]. The studying of symmetric key ciphers relates mainly to the study of stream ciphers and to their applications [3].

So, the strength of these cipher methods are depended mainly on the length of the key combination, so the work simulated this property and converted the weights numbers to be the key of the cipher system, the key length needed here will be discussed later.

The next section presents an overview of cipher system that uses the concepts of neural network for adopting the stream cipher. Then, in the next section the Encryption stage is described. After that Decryption stage is illustrated. Then the results from an experiment are presented. Finally conclusions are expressed.

Cipher System using Neural Network: An Overview:

The suggested design explains that, at the sending end the encryption process will be done using Neural Network. The sender concerned with encrypting the plaintext using feedforward neural network to generate the final ciphertext which is ready for transmitting by the sender via secure or maybe insecure channel.

At the receiving end, the decryption process will be done using feedforward neural network to obtain the plaintext.

Figure 2 shows the block diagram of the suggested design which comprises the encryption and decryption modules with their corresponding inputs and outputs.

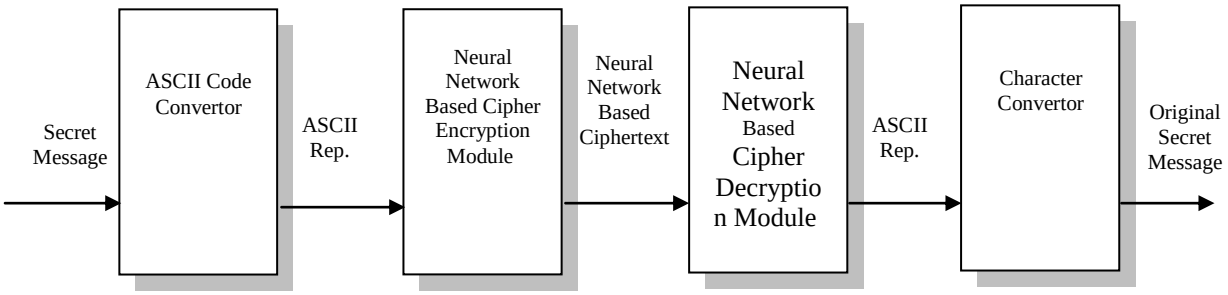

Fig 2: A Block diagram of the suggested design

Encryption Stage

As mentioned in the last section that the encryption process will be performed using neural network.

1. ASCII Code Convertor

This method converts the secret message from the plaintext to the ASCII Code representation because the Hebbian Neural Network accepts the numbers as an input,using the ASCII Code format stored in the computer.[4].

The following figure represents the block diagram of the ASCII Code Converting.

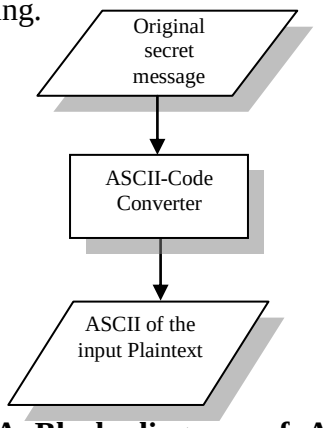

Fig 3: A Block diagram of ASCII Code Convertor

2. Neural Network Based Cipher Encryption Module (NNBCEM)

This module is the kernel of the research which represents a new technique that adopted by suggested desgin.

The purpose of NNBCEM is to improve the security of the overall system. Instead of sending ciphertext (as in traditional methods) encrypted by a method with fixed key length, the plaintext will encrypt by NNBCEM to obtain neural network based ciphertext before sending it to the receiver.

The suggested design adopts the *hebbian neural network* which will be learned by an unsupervised approach as showed in figure 4.

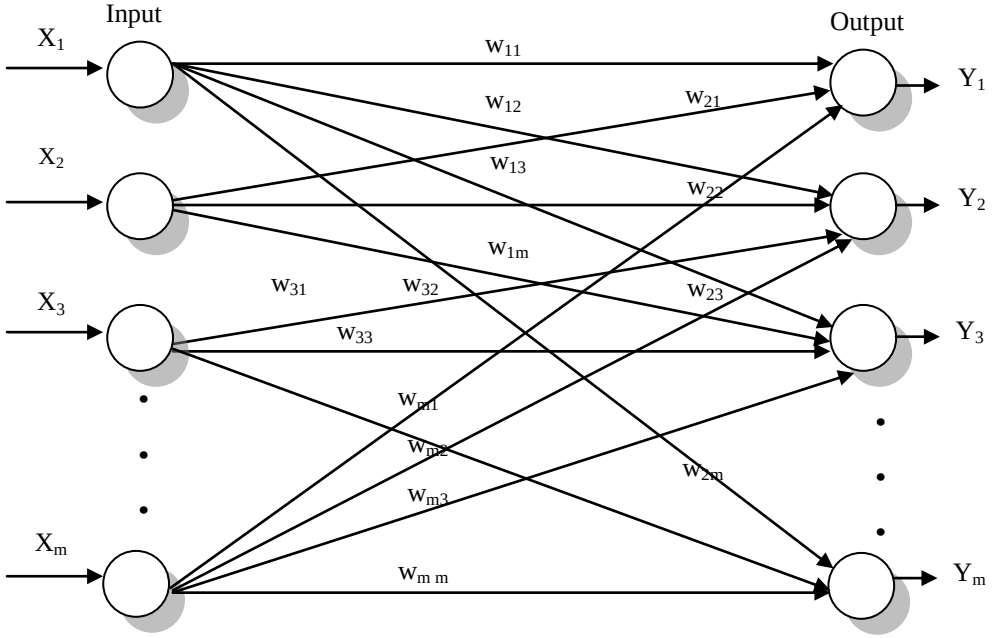

Fig 4: Architecture of the hebbian network within NNBCEM

In the suggested Neural Network The vector $X_1, X_2, \ldots, X_m$ is the input file which represents the ASCII Code of the input plaintext, while $Y_1, Y_2, \ldots, Y_m$ is the vector of the output file that represent the final ciphertext (neural network based ciphertext) that it is ready for transmitting to the receiver, and the weight represents the connection strength between the (i^{th}) input node and the (j^{th}) output node.[5][6].

So, if the message length is 10 characters, that is mean the needed is 10*10=100 different weights, i.e. 100 number which is the key length, so as the length of the key.

From figure 4 it is clearly that number of the nodes in the input and output layer equal the number of the characters in the input file (plaintext), therefore the needed is (m^2) initial weights that must be prepared before the execution of the NNBCEM.

The suggested hebbian network has the following properties

- It is a single layer (does not contain any hidden layer), this topology provides an ability of simple implementation. And fast learning speed if it is compared with another networks that contain hidden layers.
- It is feedforward network which means that the propagation of the signals in the network will be in one direction only, therefore each

neuron will depend on the directed input signals only.

- There is no activation function as in the conventional hebbian neural network. So the design overcomes the problem of selecting which is one of the activation functions that gives the better results in the learning process.

The block diagram of NNBCEM is viewed in figure 5.

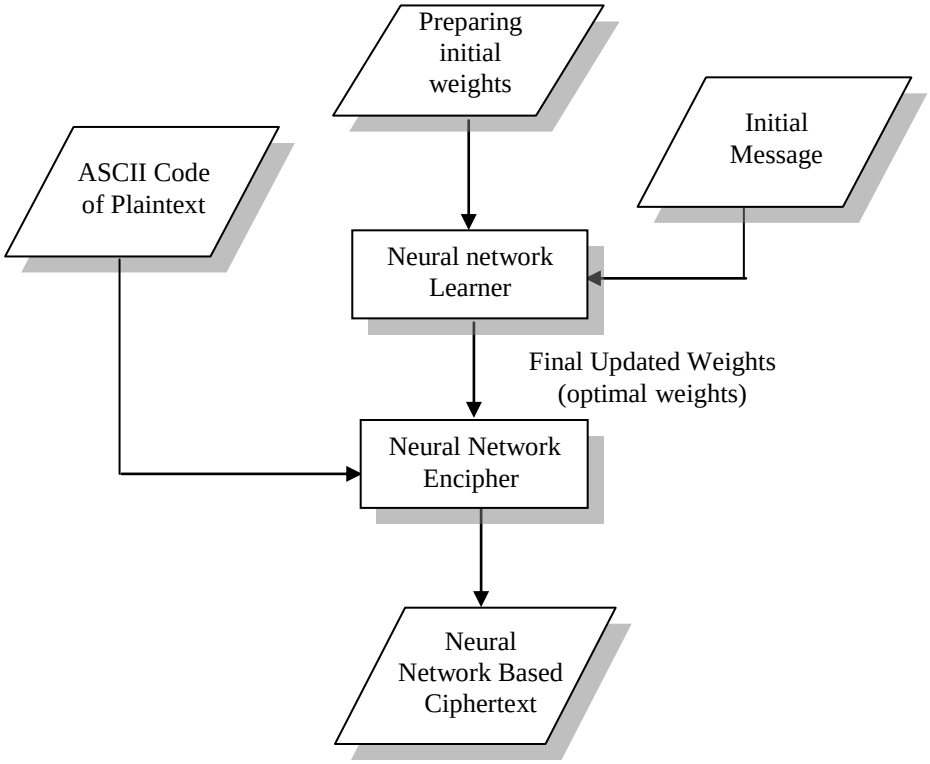

Fig 5: A Block diagram of NNBCEM

From figure 5**,** it is clearly that there are two functions within NNBCEM which are:

1. *Neural Network Learner*
2. *Neural Network Encipher*

Which will be explained in the next two subsections.

2.1 Neural Network Learner

NNBCEM performs the learning process on the weights through number of steps to find the final updated weights that will be used by the neural network encipher.

Neural network learner at first requests the sender to input any message or any file contains the desired message. So this message represents as input for the network, and message must have the same length of the plaintext(secret message).[6].

During the learning process, the neural network learner will prepare two-dimensional array of size (*m***m*), (while *m* represents the initial message size), then it will perform normalization operation for the initial weights, in order to avoid the problem of overflow in the weight values. The normalization process is implemented by applying the following formula:

$$NW_{ij} = \frac{W_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{m} W_{ij}} \quad \ldots (1)$$

This process is applied on all the initial weights to find the modified weights.

Neural network learner sub-module computes the current actual

output as initial output that is used to find the updated weights. The output (Y_j) is found by using the following formula:

$$Y_j = \sum_{i=1}^{m} NW_{ij} * X_i \quad \text{such that , } j=1, 2... m \qquad \text{… (2)}$$

Where NW_{ij} is the normalized weights and X_i represents the i^{th} element in the initial message file.

The weight updating process is performed by the following formula:

$$\Delta W_{ij} = \zeta * Y_j * \left[X_j - \sum_{k=1}^{j} NW_{ik} * Y_k \right]$$

Such that $i, j=1, 2... m$ **… (3)**

Where **(ζ)** is the learning factor (positive small value less than one), and **(ΔW_{ij})** is the updated weight.

Formula (2) and (3) at the first step works on the normalized initial weights then it will deals with the last updated weights and continuing in this process until the learning process is completed.[7].

2.2 Neural Network Encipher

Neural Network Encipher receives the plaintext and the optimal weights that are obtained from neural network learner (as showed in figure 7).

The encryption process is done by applying the formula 2 where Y_j represents the j^{th} element that will be obtained and written in the output file of neural network based ciphertext, X_i represents the ASCII of the i^{th} element within the file of stream based ciphertext, while W_{ij} is substituted by the optimal weights that are found within neural network learner.

At this stage, the neural network ciphertext is obtained and can be sent to the front end (receiver) which he/she must has the ability to decrypt the neural network ciphertext to obtain the original secret message.[8].

Decryption Stage:

1. Neural Network Based Cipher Decryption Module (NNBCDM)

This sub-module is designed to be used by the receiver to recover the plaintext from the received neural network based ciphertext.

NNBCDM uses the inverse of the optimal weights to carry out the decryption process, which means that the sender and receiver sharing the final updated weights (optimal weights) as secret keys, and then the receiver (by NNBCDM) will find the inverse of the optimal weights matrix using any numerical algorithm.

The block diagram of NNBCDM is shown in figure 8.

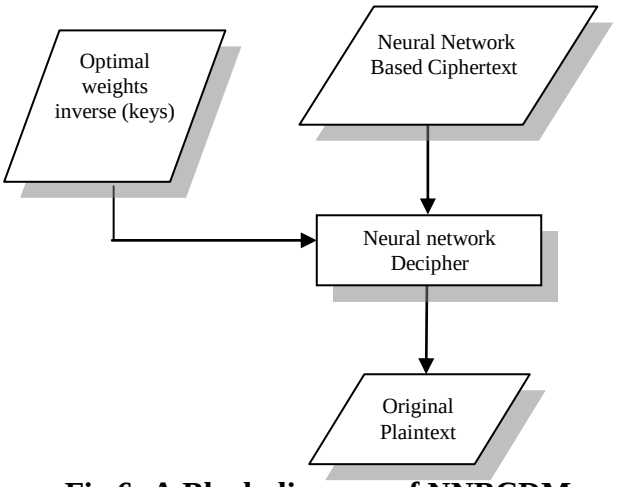

Fig 6: A Block diagram of NNBCDM

Neural network decipher implement the decryption process by implementing formula (4).

$$X_j = \sum_{i=1}^{m} (IW)_{ij} * Y_i \quad \text{Such that, } j=1, 2 \ldots m \qquad \ldots(4)$$

Where Y_i is the ASCII code of the i^{th} element in the input neural network based ciphertext file, IW_{ij} is the inverse weight that represents the connection strength between the input node i and the output j in the adopted neural network i.e. between the i^{th} code in the neural network based ciphertext file and the j^{th} code in the plaintext, and X_j represents the j^{th} code of the plaintext that represent the output of NNBCDM.

2. Character Convertor

This module uses the principles of the reverse ASCII Code Table in order to obtain the original message.

The following figure represents the block diagram of this operation.

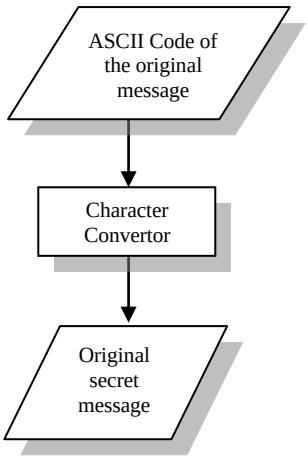

Fig 7: A Block diagram of Character Convertor

Finally after that explain the main modules that are implemented in the system separately, and will present two block diagrams that show the suggested system design in details.

The first block diagram (figure 8) shows the system from the sending point of view and the second block (figure 9) diagram shows the system from the receiver point of view.

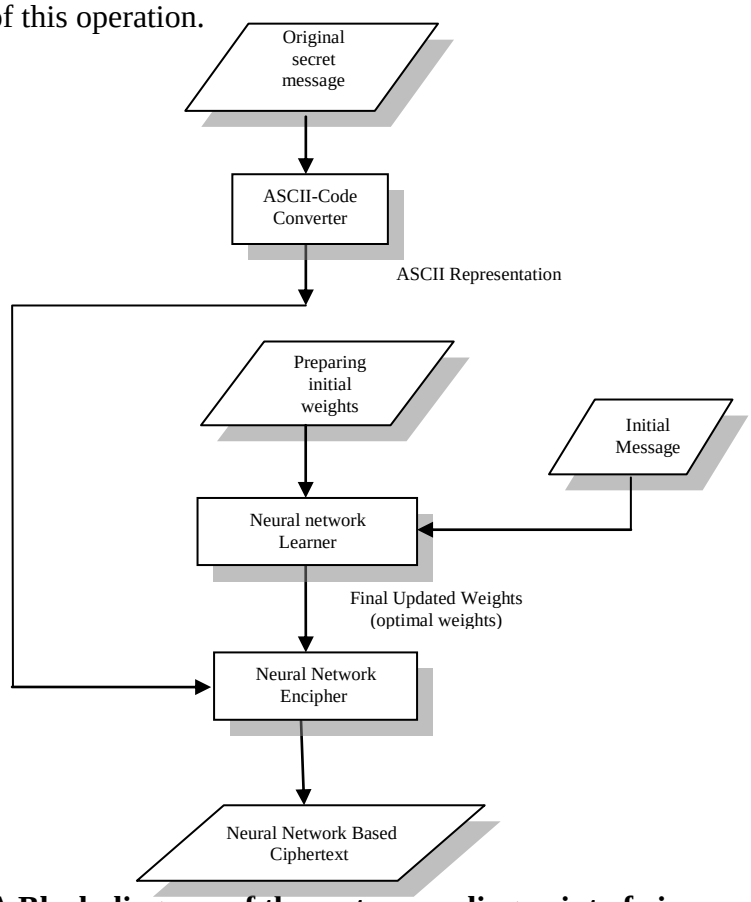

Fig 8: A Block diagram of the system sending point of view

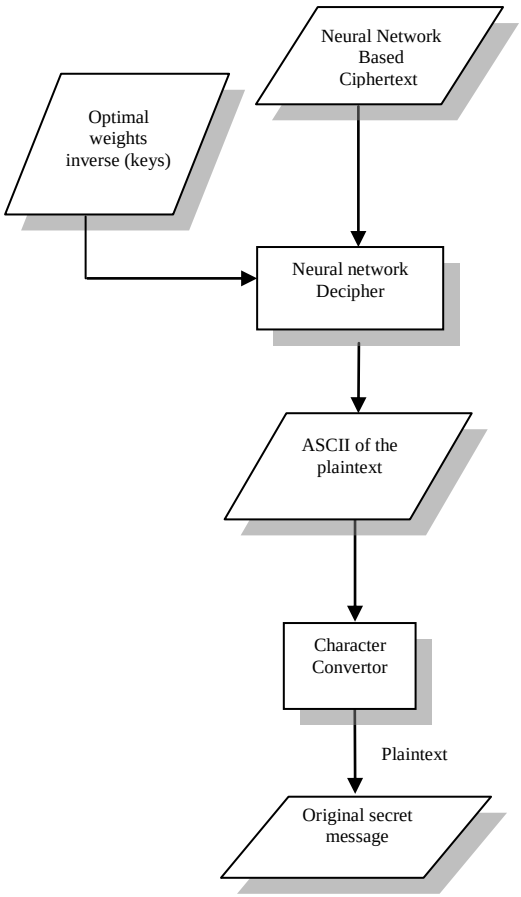

Fig 9: Block diagram of the system receiver point of view

Results:

This section testing the system stages including their modules by selecting an arbitrary message that represents secret information and will show the encryption process and the decryption process according to figure 11 and 12 in order to show the practical side of this work. Visual Basic.net version 2005 was used for execution of the work and implementing the results.

1. Explaining the Implemented Results

This section testing the proposed system, step by step to show the processes of the work, where the first stage is Encryption process including ASCII-Code Converter module that take the plaintext or secret message which is **"We Modify the Hebbian Neural Network for Text Encryption Method"** be converted to ASCII-Code using ASCII-Code Converter (as in figure 10), which will be a set of numbers equal to the length of the secret message (63 characters). And the NNBCEM module in which the sender will transmit the characters of the resulted ASCII codes to the NNBCEM (Neural network cipher), which has two functions Neural Network Learner and Neural Network Encipher.

The Neural Network Learner function accepts the initial message which is "This is the key in modified Neural Network for Text Encryptions" with initial weights as inputs and output the optimal weights (as in figure 11). Then Neural Network Encipher function take the optimal weights that outputs from previous function with the characters resulted from ASCII-Code Converter, the resulted ciphertext (as an integer numbers) from the neural network based ciphertext (as in figure 11), will be sending to the receiver, which must implements NNBCDM (Neural network Decipher) to recover the plaintext.

The second stage is decryption process which include the NNBCEM module and Character Convertor module. NNBCDM (Neural network Decipher) uses the inverse of the optimal weights with the ciphertext (as an integer numbers) resulted from Encryption process to carry out the first module decryption process, then Character Convertor module take the resulted ASCII Code for letters as input by reversing ASCII Code Table of the numbers to obtain the original message **"We Modify the Hebbian Neural Network for Text Encryption Method"**(as in figure 12).

which means that the sender and receiver sharing the final updated weights (optimal weights) as secret keys.

2. The Implemented Example

Now consider the secret message is:
" We Modify the Hebbian Neural Network for Text Encryption Method".

It is clearly that the length of this message is 63 characters, this means 63*63 key length, so you can imagine how complex the key is.

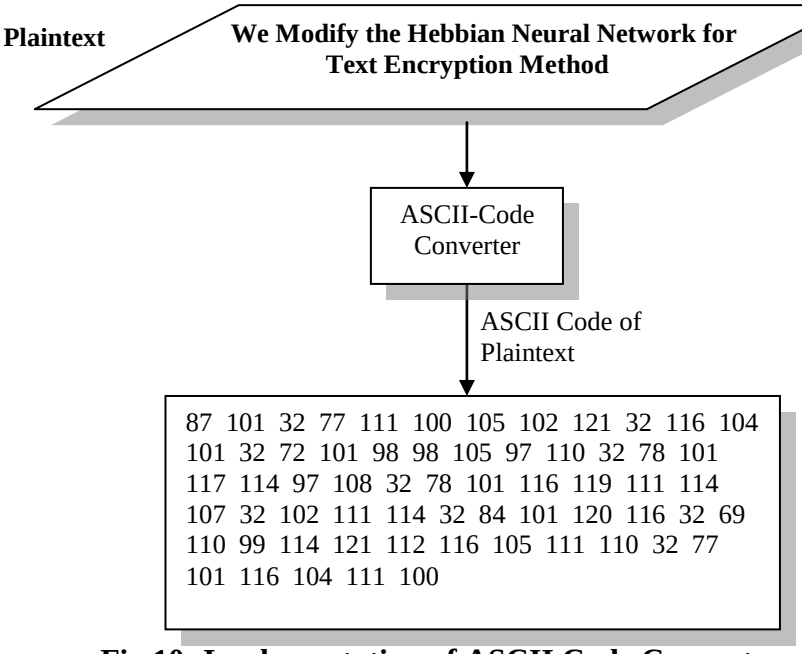

Fig 10: Implementation of ASCII Code Convertor

It is clearly that the length of ASCII Code Numbers equal to the length of the secret message (63 characters).

Note that each word is followed by a space and that is not a weakness because the suggested Neural Network produces different cipher code for each character even if this character is repeated and this is an advantages because the intruder will assumes that the repeated codes are the same letter and on discovering the ciphertext he will find no repeated codes.

The sender will transmit the characters of the resulted ASCII codes to the NNBCEM.

Implementation of NNBCEM is shown in figure 11.

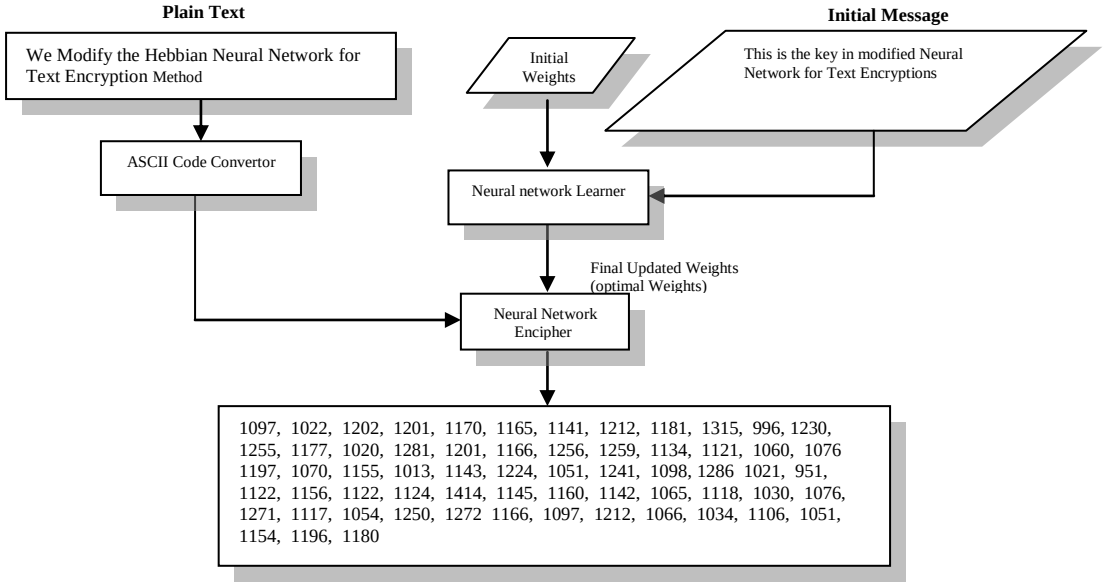

Fig 11: Implementation of NNBCEM

After implementing NNBCEM, the sender will obtain the neural network based ciphertext, which will be sending to the receiver.

After the implementation the NNBCEM in the last figure, the ciphertext (as an integer numbers) will be sending to the receiver. The receiver must implements NNBCDM to recover the plaintext.

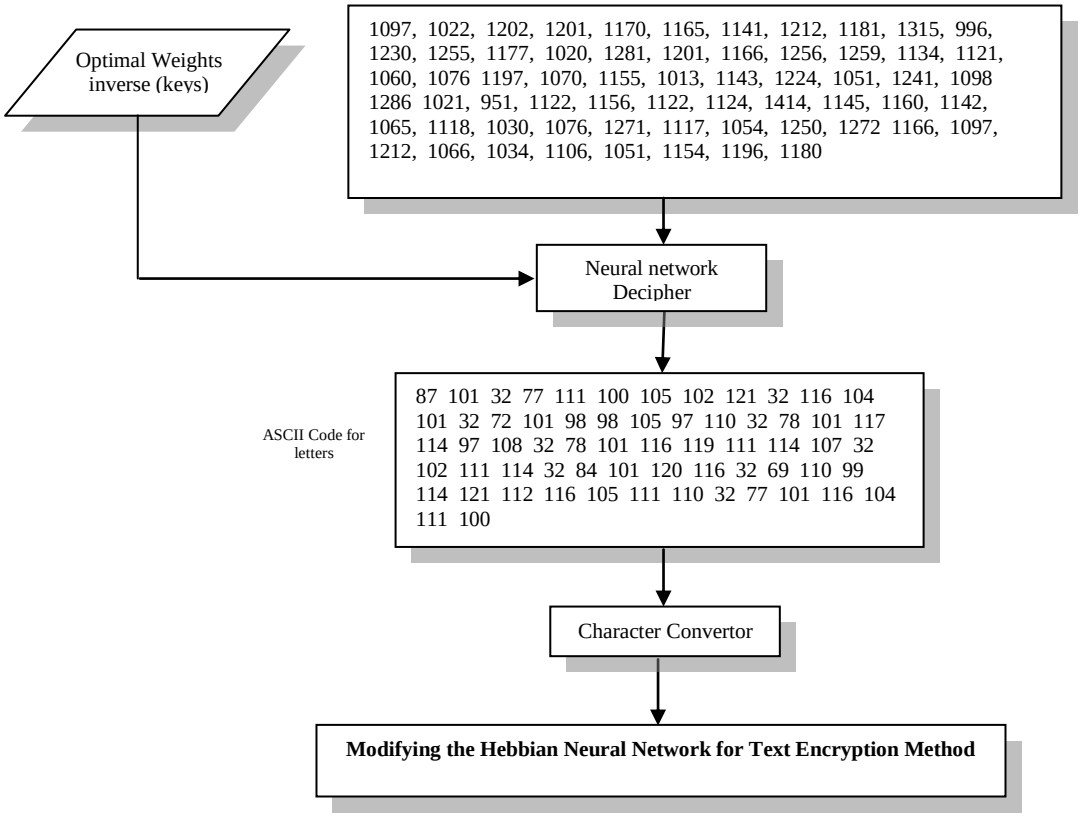

Fig 12: Implementation of NNBCDM

After the receiver implements the NNBCDM and he/she recovers the plaintext, it is clearly that the NNBCEM and NNBCDM are with good performance, because the plaintext is recovered within NNBCDM and it is identical to the original plaintext that is forwarded to the receiver.

Conclusions:

This paper introduced the modified cipher system that use the concepts of neural network which is provides high security. In this work, modernization of the Hebbian Neural Network with no Activation Function so the secret message will be encrypted by unsupervised neural network method. The results demonstrate that the decryption process is performed with good results because the original message has been successfully recovered during the decryption process. And achieved the goal because the intruder can not break the transmitted ciphertext because the ciphertext is ciphered using a very long key.

References:

1. Zurada, J. M., 1996, "Introduction to Artificial Neural Systems", Jaico

Publishing House, Second Edition, 223-226.
2. Jain Mao, A. K., J. 1996, "Artificial Neural Networks: A Tutorial", IEEE Computer Society. 29(3): 31-44.
3. Mandic, D. and Chambers, J. 2001: "Recurrent Neural Networks for Prediction: Learning Algorithms, Architectures and Stability", John Wiley & Sons, First Edition, New York.
4. Kinnebrock, W., 1995, "Neural Networks, Fundamentals, Applications, Examples", Technical University Rheinland-Pfalz, Second Revised Edition, 115-118.
5. Denning D. E., 1982, "Cryptography and Data Security", Addison Wesley.
6. Henk C.A., Tilborg V., 2000, "An Introduction to Cryptology", Kluwer Academic Publishers, Fourth Edition, 127-134.
7. Mogollon, M., 2007, "Cryptography and Security services: Mechanisms and applications ", IGI Globel, Second Edition. 54-62.
8. Schneier, B., 2006, "Applied Cryptography: Protocols, Algorithms and Source Code", John Wily & Sons Inc., Third Edition, 215-217.

تعديل شبكة هابين لتشفير نص

نور عدنان إبراهيم*

*جامعة بغداد \ كلية العلوم للبنات \ قسم علوم الحاسبات

الخلاصة:

كثيرا ما نشاهد وجود طرائق جديدة لكسر النص المشفر المنقول بين طرفين نتيجة لاستعمال طرائق معينة من اجل معرفة مفتاح التشفير, لذا فان هدف هذا البحث هو بناء نظام لارسال و استلام البيانات من دون القدرة على كسر النص المنقول من المتطفلين(الدخلاء).

اذ ان الطريقة المقترحة هي تحديث في شبكة عصبية لاجل القيام بعملية التشفير.

من المعروف ان امنية اي طريقة التشفير تعتمد على امنية المفتاح المستخدم للتشفير و كذلك طول هذا المفتاح, فلذلك قمنا باستعمال الأوزان الناتجة من عملية التعليم بوصفه مفتاحا للتشفير وفك الشفرة, وبذلك نلاحظ, فلذلك المفتاح طويل جدا و لا يمكن تفسيره او كسره.