General Regression Neural Network Application for Dynamic Data Compression and Decompression

Safwan Omar Hasoon **Susan Hassan Mohammed**
Dr.safwan1971@yahoo.com
College of Computer Sciences and Mathematics
University of Mosul

Received on:18/9/2010 **Accepted on:10/11/2010**

ABSTRACT

In the last decade have been witnessed a great development in artificial intelligence especially in neural network .This paper have been employed neural networks techniques for data compression and decompression.

The application of General Regression Neural Network (GRNN) in data compression is very important of data transmission, because this technique offers less than memory storage and time for transferring of the data over computer networks or internet. Taking into consideration the data compression provides security of these data.

The matlab version (R2009a) is used for designing the propose system of neural network (GRNN) to dynamic data compression and decompression .

Keywords: Neural Network, Data Compression, Data Decompression, GRNN

الملخص

شهد العقد الاخير تطور كبير في مجال الذكاء الاصطناعي وخصوصا في الشبكات العصبية .في هذا البحث تم تطبيق تقنية الشبكات العصبية لكبس وفك كبس البيانات.

ان تطبيق GRNN لكبس البيانات مهم جدا في مجال نقل البيانات لان هذه التقنية تقدم ذاكرة خزنية ووقت اقل لنقل البيانات في شبكات الحاسبات او الانترنت. اخذين بنظر الاعتبار ان كبس البيانات يعطي امنية لهذه البيانات.استخدمت لغة matlab ver(R2009a) لتصميم النظام المقترح للشبكة العصبية GRNN لغرض كبس البيانات وفكها .

1. Introduction:

Data compression is an effective means for saving storage space and network bandwidth. A large number of compression schemes have been devised based on character encoding or on detection of repetitive strings [10,3], Many compression schemes achieve data reduction rates to 2.3−2.5 bits per character for English text [5].
In many situations arising in digital communications and data processing, the encountered strings of data display various structural regularities or are otherwise subject to certain constraints, thereby allowing for storage and time-saving techniques of data compression.

Given a discrete data source, the problem of data compression is first to identify the limitations of the source, and second to devise a coding scheme which, subject to certain performance criteria, will best compress the given source. When no a priori knowledge of the source characteristics is available, and if statistical tests are either impossible or unreliable, the problem of data compression becomes considerably more

complicated. In order to overcome these difficulties one must resort to universal coding schemes whereby the coding process is interlaced with a learning process for the varying source characteristics. Such coding schemes inevitably require a larger working memory space and generally employ performance criteria that are appropriate for a wide variety of sources [7].

The use of compression for storing text files has become inherent part of personal as well as commercial computing. The various compression applications available perform two functions, compression and decompression. The text document is first compressed and then the entire document is decompressed when required [1,8]. This has some implications such as the unnecessary use of disk space for storing the compressed document as well as uncompressed document at the same time. Another implication is that even though an end user may require only a part of the document, the entire document as a whole is decompressed [1,4].

2. Neural Networks:

Neural network are simplified models of the biological neurons system .the human brain computes in an entirely different way from digital computer . The brain is a highly complex, nonlinear, and parallel information processing system .it has ability to organize its structural constituents, known as neurons or nerve cells. A neural network is a massively parallel distributed processors made up of simple processing units, which has a natural capability for storing experimental knowledge and making it available for use.

Above all an Artificial Neural Network (ANN) is an information processing system that is inspired by the way biological nervous systems, such as the brain , process information . The key element of this paradigm is the novel structure of the information processing system . It is composed of a large number of highly interconnected processing elements (neurons) working in unison to solve specific problems. ANNs, like people, learn by example . An ANN is configured for a specific application , such as pattern recognition or data classification , through a learning process . Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons . This is true of ANNs as well .

To get the correct output training should require. This process is known as learning various kinds of learning processes are there . Namely supervised learning unsupervised learning methods are very popular . In supervised learning a teacher should be present were as in unsupervised learning has to be performed without teacher[11].

2.1 General Regression Neural Network (GRNN):

The General Regression Neural Network (GRNN) is a neural network architecture that can solve any function approximation problem. The learning process is equivalent to finding a surface in a multidimensional space that provides a best fit to the training data, with the criterion for the “best fit” being measured in some statistical sense. The generalization is equivalent to the use of this multidimensional surface to interpolate the test data.

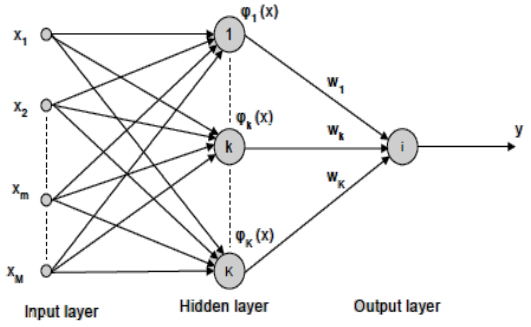


Figure (1) General regression neural network

Figure (1) is the overall network topology implementing the GRNN. As it can be seen from the figure, the GRNN consists of three layers of nodes with entirely different roles:

1. The input layer, where the inputs are applied.
2. The hidden layer, where a nonlinear transformation is applied on the data from the input space to the hidden space; in most applications the hidden space is of high dimensionality.
3. The linear output layer, where the outputs are produced.

The most popular choice for the function φ is a multivariate Gaussian function with an appropriate mean and auto covariance matrix. The outputs of the hidden layer units are of the form .

$$\varphi_k[x] = \exp\left[-(x - v_k^x)^T (x - v_k^x/(2\sigma^2)\right] \quad \ldots(1)$$

When $\mathbf{V}_k^x$ are the corresponding clusters for the inputs and $\mathbf{V}_k^y$ are the corresponding clusters for the outputs obtained by applying a clustering technique of the input/output data that produces K cluster centers [2].

$\mathbf{V}_k^y$ is defined as :

$$v_k^y = \sum_{y(p) \in \text{cluster } k} y(p) \quad \ldots(2)$$

N_k is the number of input data in the cluster center k, and

$$d(x, v_k^x) = (x - v_k^x)^T (x - v_k^x) \quad \ldots(3)$$

With

$$v_k^x = \sum_{x(p) \in \text{cluster } k} x(p) \quad \ldots(4)$$

The outputs of the hidden layer nodes are multiplied with appropriate interconnection weights to produce the output of the GRNN. The weight for the hidden node k (i.e., w_k) is equal to

$$W_k = \frac{v_k^y}{\sum_{k=1}^{k} N_k \exp\left[-\frac{d(x, v_k^x)^2}{2\sigma^2}\right]} \quad \ldots(5)$$

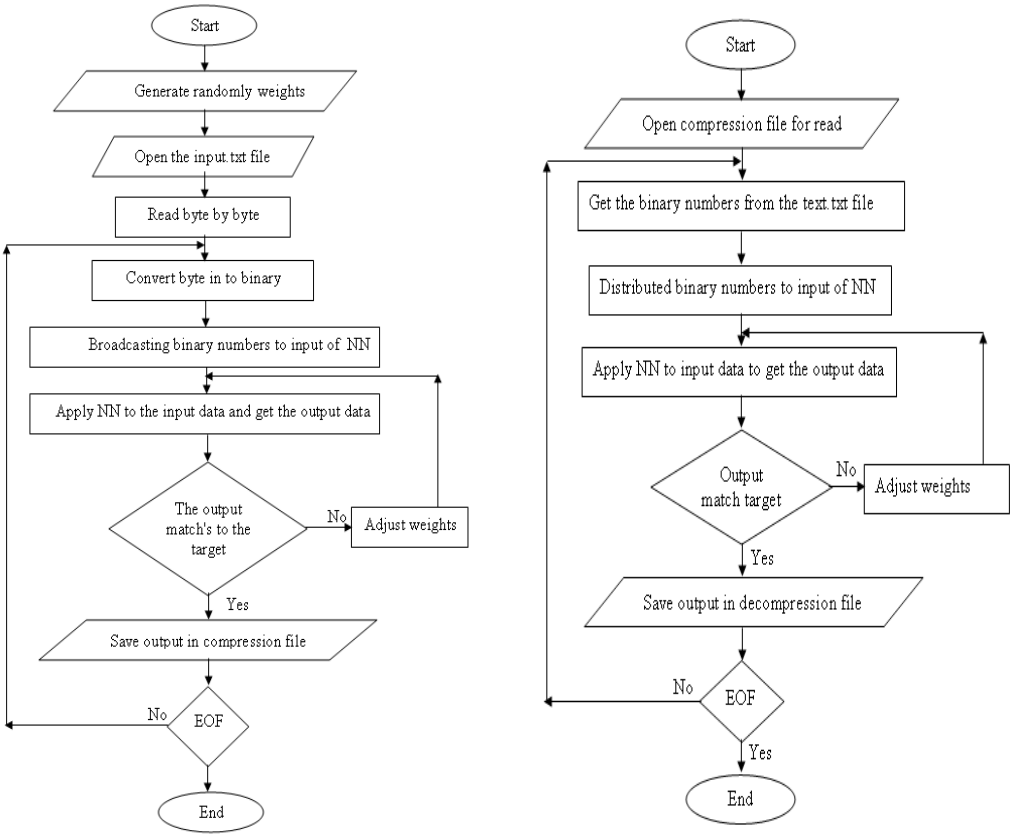
The selection of an adequate set of training examples is very important in order to achieve good generalization properties. The set of all available data is separated in two disjoint sets: training set and test set. The test set is not involved in the learning phase of the networks and it is used to evaluate the performances of the models [9]. The configuration of the neural network model is determined by the nature of the problem to be solved. The dimension of the input vector used defines the number of inputs neurons [6].

3. Application of the GRNN to Data Compression:

The steps of the GRNN data compression and decompression algorithm as show below:

1. Open text file for reading
2. Read character by character and convert each character to ASCII
3. Convert ASCII to binary number
4. Broadcasting the binary number to the input layer of GRNN
5. Applied training of the neural network to get the output throws the hidden layer using equations 1 to 5.
6. Compare the result of the GRNN with the target, if match then stop; otherwise adjust the weights to continue the learning of the neural network by using steps 5, 6 again.

The figure (2) explain the process of data compression and decompression using neural network respectively



(a) (b)

Figure (2) a- GRNN Comp. flowchart and (b)- GRNN Decom. flowchart

The main objective of this paper is to design hybrid system between neural network with data compression and decompression. The common between two techniques is to offer application support efficiently solving this problem because combine the two techniques is to provide less than memory storage and time process to transfer data over channel.

GRNN can be used to compress a character by performing training (the training input vector and target vector shown in table 1). the number of nodes in output layer can be determined dynamically depended on the order of the character in alphabetical and binary target pattern, when A equal the value 1, B equal the value 2, and so on until the Z equal the value 26 see table 1. This facility to support the proposed system, because offers less than memory storage for the data. The input layer has 8 nodes to implement

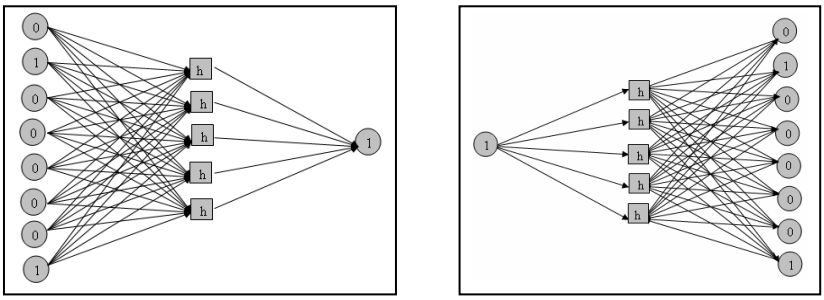
an ASCII character wish to compress. The number of nodes in hidden layer is given smaller for compression.

Table (1). Input and target pairs of the GRNN

Input pattern	Binary Target Pattern	Decimal Target pattern	No. of nodes in output layer depended on the No. bits in binary target
A (01000001)	1	1	1
B (01000010)	10	2	2
C (01000011)	11	3	2
D (01000100)	100	4	3
E (01000101)	101	5	3
F (01000110)	110	6	3
G (01000111)	111	**7**	3
H (01001000)	1000	8	4
I (01001001)	1001	9	4
J (01001010)	1010	10	4
K (01101011)	1011	11	4
L (01001100)	1100	12	4
M (01001101)	1101	13	4
N (01001110)	1110	14	4
O (01001111)	1111	15	4
P (01010000)	10000	16	5
Q (01010001)	10001	17	5
R (01010010)	10010	18	5
S (01010011)	10011	19	5
T (01010100)	10100	20	5
U (01010101)	10101	21	5
V (01010110)	10110	22	5
W (01010111)	10111	23	5
X (01011000)	11000	24	5
Y (01011001)	11001	25	5

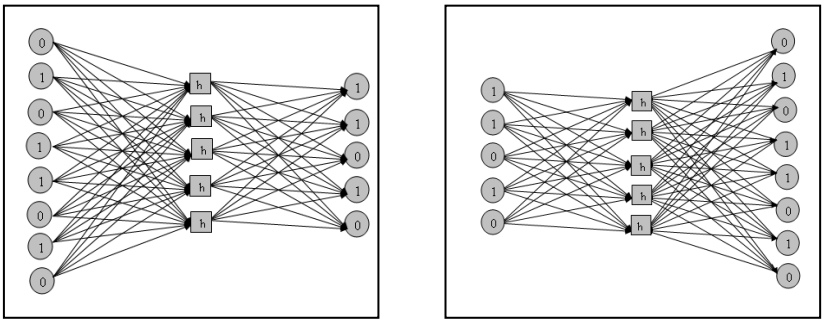
Z (01011010)	11010	26	5

The structure of the proposed system for compression and decompression for example characters 'A' and 'Z' are shown in figure(3) and figure (4) respectively.



(a) (b)

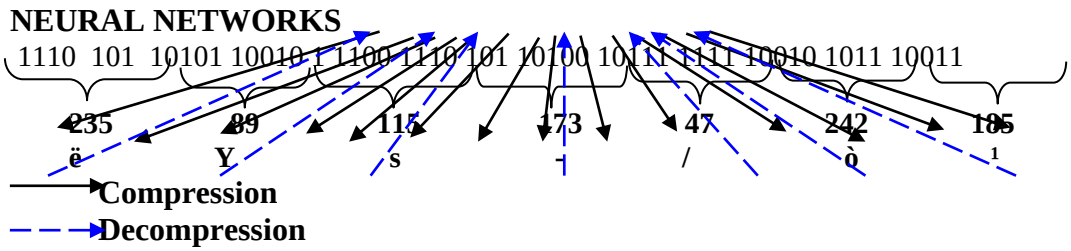
Figure (3)-(a) Compression & (b) decompression for (A) character using GRNN



(a) (b)

Figure (4)-(a) Compression & (b) decompression for (Z) character using GRNN

The following text compression using proposed system :



The result of GRNN compression is (ë ,Y, s , - , /, ò, ¹), the number of the characters in above text are 14 byte, while the numbers of the character in the text after compress operation are 8 byte, therefore the data reduction of this system to 4 bit per character.

4. Conclusion:

In this paper a new idea to text compression based on neural network, the GRNN technique used to data compression and decompression. This technique for proposed system offers less than memory storage and time-consuming comparing with the traditional text compression. The average data reduction rate of the proposed system to 4 bit per character, while the traditional text compression data reduction rates to 2.3-2.5 bit per character is stated in introduction section.

The proposed system run very fast because the nodes in neural networks are work together in highly parallel process.The whole system is flexible (i.e. the output of the nodes for GRNN compression is determined dynamically), this facility is provides to extend for compression small letters.

Finally the system can be used for real time applications, computer networks and internet.

REFERNCE

[1]. A A. Kamath, A. Kant , A. Srivatsa and Harisha J.A , *Dynamic Decompression for Text Files,World Academy of Science*, Engineering and Technology 59 2009.

[2]. C. Christodoulou, M. Georgiopoulos, *Applications of Neural Networks in Electromagnetics*, Artech House, 2001 .

[3]. D. A. Lelewer and D. S. Hirschberg, *Data Compression*, ACM Computing Surveys 19, 3 (September 1987), 261.

[4]. F. Awan, N. Zhang, N. Motgi, R. Iqbal, and A. Mukherjee, *LIPT: A versible Lossless Text Transform to Improve Compression Performance*, Proceedings IEEE Data Compression Conference, pp. 481-210, 2001.

[5]. G. Graefe and L. D. Shapiro, *Data Compression and Database Performance* . ACM/IEEE-CS Symp. on Applied Computing, Kansas City, MO, April 1991.

[6]. Ileana P., Athanasios K. , Philip C. , Ioan N., *Application of General Regression Neural Networks for Path Loss Prediction* , 2003 .

[7]. J. ZIV AND A. LEMPEL, *A Universal Algorithm for Sequential Data Compression* , IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-23, NO. 3, MAY 1977 .

[8]. Marc L. Corliss , E. Christopher Lewis , Amir Roth, *The implementation and evaluation of dynamic code decompression using DISE*, ACM Transactions on Embedded Computing Systems (TECS), v.4 n.1, p.38-72, February 2005.

[9]. S. Haykin, *Neural Networks. A Comprehensive Foundation*, IEEE Press, McMillan College Publishing Co., 1994 .

[10]. T. Bell, I. H. Witten and J. G. Cleary, *Modeling for Text compression*, ACM Computing Surveys 21, 4 (December 1989), 557.

[11]. T.N.Shankar, *Neural network university Science Press*, 2008.