International Journal of Advances in Engineering Research http://www.ijaer.com

(IJAER) 2013, Vol. No. 5, Issue No. I, January **ISSN: 2231-5152**

IMPORTANCE AND USES OF HOPFIELD NEURAL NETWORK AND PATTERN STORAGE

Sushma Sharma

ABSTRACT

What is actually stored in practice is the information in the pattern itself. The approximation is measured in the terms of some distance, like Hamming distance. The distance feature is automatically realized through the threshold (binary) feature of the output function of the processing unit [35]. A feedback network consisting of non-linear processing units accomplishes the pattern storage. Researchers have combined neural network and genetic algorithms in number of different ways [40][41]. Schaffer et al has noted that this combination can be classified into two different ways – Supportive and Collaborative. In supportive combination, the neural network and genetic algorithms are applied simultaneously while in collaborative approach; both are integrated into a single system in which a population of neural networks is evolved to find the optimal neural network solution [42].

Evolution in ANNs has been introduced roughly at three different levels: connection weights; architectures; and learning rules. The evolution of connection weights introduces an adaptive and global approach to training, especially in the reinforcement learning and recurrent networks learning paradigm where gradient-based training algorithms often experience great difficulties.

Key Words: *Supportive, Collaborative, weights; architectures; learning.*

INTRODUCTION

In the previous chapter we have discussed details of Neural Networks (NNs) and Genetic Algorithms (GAs). We have seen that both, NNs and GAs, are based on the models from nature.

The evolution of architectures enables ANNs to adapt their topologies to different tasks without human intervention and thus provides an approach to automatic ANN design as both connection weights and structure can be evolved. The evolution of learning rules can be regarded as a process of "learning to learn" in ANNs where the adaptation of learning rules is achieved through evolution. It can also be regarded as an adaptive process of automatic discovery of novel learning rules.

Weight training in ANNs is usually formulated as minimization of an error function, such as the mean square error between target and actual outputs averaged over all examples, by iteratively adjusting connection weights. Most training algorithms, such as BP and conjugate gradient algorithms are based on gradient descent. There have been some successful applications of BP in various areas [43]-[45], but BP has drawbacks due to its use of gradient descent. It often gets trapped in a local minimum of the error function and is incapable of finding a global minimum if the error function is multimodal and/or non-differentiable.

One way to overcome gradient-descent-based training algorithm's shortcoming is to adopt hybrid evolutionary systems i.e., to formulate the training process as the evolution of connection weights in the environment determined by the architecture and the learning task. Evolutionary algorithms can then be used effectively in the evolution to find a near-optimal set of connection weights globally without computing gradient information. The fitness of an ANN can be defined according to the different needs. Two important factors, which often appear in the fitness (or error) function, are the error between target and actual outputs and the complexity of the ANN. Because EAs can address cases in real world like large, complex, non-differentiable, multimodal spaces, considerable research has been conducted on the evolution of connection weights [46] – [81].

REVIEW OF LITERATURE

The evolutionary training approach is attractive because it can handle the global search problem better. It does not depend on gradient information of the error (or fitness) function and thus is particularly appealing when this information is unavailable or very costly to obtain or estimate. The same EA can be used to train many different networks regardless of whether they are feed-forward, recurrent [54], [66], [82] – [84], higher order [60], [61] or fuzzy ANNs [69]. The general applicability of the evolutionary approach saves a lot of human efforts in developing different training algorithms for different types of ANNs. EA's are generally much less sensitive to initial conditions of training. They always search for a globally optimal solution, while a gradient descent algorithm can only find a local optimum in a neighborhood of the initial solution. . The results of research show that there is no clear winner in terms of the best training algorithm. The best one is always problem dependent. In general, hybrid algorithms tend to perform better than others for a large number of problems.

MATERIAL AND METHOD

Our objective is to store a given set of patterns so that any one of the patterns can be recalled exactly when an approximate or corrupted version, due to noise and distortion, of the pattern is presented to the network. What is actually stored in practice is the information in the pattern itself. The approximation is measured in the terms of some distance, like Hamming distance. The distance feature is automatically realized through the threshold (binary) feature of the output function of the processing unit [35]. A feedback network consisting of non-linear processing units accomplishes the pattern storage.

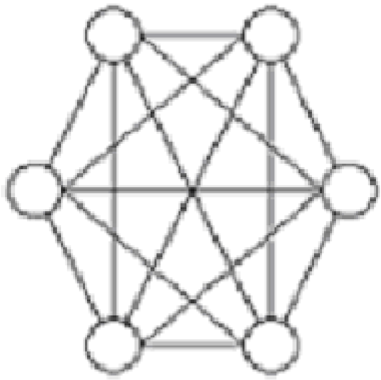

Figure 3: *Hopfield network structure with six units*

Popularized by Nobel Laureate John Hopfield, the Hopfield model is a fully connected feedback network with symmetric weights (i.e. $w_{ij} = w_{ji}$), which possess a rich class of dynamics characterized by the existence of several stable states each with its own basin of attraction [33][36][37][38]. These basins of attraction in the energy landscape tend to be the regions of stable equilibrium states [39]. The number of basins of attraction in the energy landscape depends only on the network i.e. the number of processing units and their interconnection strengths (weights). When the number of patterns to be stored is less than the number of basins of attraction, i.e. stable state, then there will be spurious stable states, which do not correspond to any desired patterns. In such a case, when the network is presented with an approximate pattern for recall, the activation dynamics may eventually lead to a stable state that may correspond to one of the spurious state or a *false energy minimum*, or to one of the stable states corresponding to some other pattern. In the latter case, there will be an undetected error in the recall. The average probability of error depends on the energy values of the stable states corresponding to the desired patterns, and the relative locations of these states in the state space, measured in terms of some distance criterion.

Keeping in mind the McCulloch-Pitts model for the units of a feedback network, where the output of each unit is fed to all the other units with weights w_{ij}, let the output function of each of the units be bipolar so that

$$s_i = f(x) = \operatorname{sgn}(x_i) \tag{1}$$

and

$$x_i = \sum w_{ij} s_i - \theta_i \tag{2}$$

where θ_i is the threshold for the unit *i*. We will assume $\theta_i = 0$ for convenience. The state of each unit is either +1 or –1 at any given instant of time. Due to feedback, the state of a unit depends on the states of the other units. The updating of the state of a unit can be done synchronously or asynchronously. In the synchronous update all the units are simultaneously updated at each time instant, assuming that the state of the network is frozen until update is made for all the units. In the asynchronous update a unit is selected at random and its new state is computed. Another unit is selected at random and its state is updated using the current state of the network. The updating using the random choice of a unit is continued until no further change in the state takes place for all the units. That is, the state at time *(t+1)* is the same as the state at time t for all the units. That is,

$$s_i(t+1) = s_i(t), \qquad \text{for all } i \tag{3}$$

In this situation, we can say that the network activation dynamics reached a stable state. We assume asynchronous update for future because asynchronous update ensures that the next state is at most unit Hamming distance from the current state.

If the network is to store a pattern **a** = $(a_1, a_2, \ldots, a_N)^T$, then in a stable state we must have the updated state value to be the same as the current state value. That is,

$$\operatorname{sgn}\left(\sum_{j=1}^{N} w_{ij} a_j\right) = a_i, \text{ for all } i \tag{4}$$

This can happen if $w_{ij} = (1/N) a_i a_j$, because

$$\sum_{j=1}^{N} w_{ij} a_j = \frac{1}{N}\sum_{j=1}^{N} a_i a_j a_j = \frac{a_i}{N}\sum_{j=1}^{N} a_j^2 = a_i \tag{5}$$

where $a_j^2 = 1$ for bipolar (± 1) states.

$$w_{ij} = \frac{1}{N}\sum_{l=1}^{L} a_{li} a_{lj} \tag{6}$$

then the state a_k will be stable if

$$\operatorname{sgn}\left(\frac{1}{N}\sum_{j=1}^{N}\sum_{l=1}^{L} a_{li} a_{lj} a_{kj}\right) = a_{ki}, \text{ for all } i \tag{7}$$

Taking out the l = k term in the summation and simplifying it by using $a_{kj}^2 = 1$, we get,

$$\operatorname{sgn}\left(a_{ki} + \frac{1}{N}\sum_{j=1}^{N}\sum_{l \neq k} a_{li} a_{lj} a_{kj}\right) = a_{ki}, \text{ for all } i \tag{8}$$

Since $a_{ki} = \pm 1$, the above is true for all a_{ki}, provided the cross term in above equation does not change the sign of a_{ki} plus the cross term.

(IJAER) 2013, Vol. No. 5, Issue No. I, January **ISSN: 2231-5152**

CONCLUSION

Till now, we are through with preliminaries of pattern recognition, neural networks, genetic algorithms and one specific model of ANN i.e. Hopfield model by and large used for storing (memorizing the patterns). In the next chapter, we will try to focus our discussion on formation of hybrid evolutionary systems by combining the genetic algorithms with Hopfield model of ANN. This combination, when applied to different pattern recognition problems, gives better results compared to ANN or GA alone.

REFERENCES

[1] Ross P E, "Flash of Genius", *Forbes*, pp. 98-104, November 1998

[2] Watanabe S, "*Pattern Recognition: Human and Mechanical*", New York: Wiley, 1985

[3] Dominic S, Das R, Whitley D, and Anderson C, "Genetic reinforcement learning for neural networks", in *Proceedings of IEEE Int. Joint Conf. Neural Networks (IJCNN'91 Seattle)*, vol. 2, pp. 71-76,1991

[4] Devroye L, Gyorfi L and Lugosi G, *A Probabilistic Theory of Pattern Recognition, Berlin*: Springer-Verlag, 1996

[5] Kosko, B., "Neural Networks and Fuzzy Systems" Prentice-Hall India,2005

[6] Fu K S, *Syntactic Pattern Recognition and Applications*, Englewood Cliffs, NJ: Prentice Hall, 1982

[7] Desai M S, *Noisy pattern retrieval using associative memories*, MSEE thesis, University of Louisville, Kentucky, 1990

[8] Stanley S M, "A Theory of Evolution above the Species Level", *Proceedings of National Academy of Sciences*, vol 72, pp. 646450, 1975

[9] Bremermann H J, Rogson M, "An Evolution Type Search Method for Convex Sets", *ONR Technical Report*, Contract 222(85) and 3656(58), UC Berkley, 1964

[10] Yan W, Zhu Z, and Hu R, "Hybrid genetic/BP algorithm and its application for radar target classification", in *Proc. 1997 IEEE National Aerospace and Electronics Conf., NAECON. Part 2 (of 2)*, pp. 981-984, 1997

[11] Yao X, "Evolving Artificial Neural Network", *Proceedings of the IEEE*, vol. 87, number 9, September 1999

[12] Homaifar A and Guan S, "Training weights of neural networks by genetic algorithms and messy genetic algorithms", in *Proceedings of 2nd IASTED Int. Symp. Expert Systems and Neural Networks*, M. H. Hamza, Ed. Anaheim, CA: Acta, pp. 74-77, 1990

[13] Prados D L, "New learning algorithm for training multi-layered neural networks that uses genetic-algorithm techniques", *Electron. Lett.*, vol. 28, pp. 1560-1561, July 1992

[14] Cordella L P, Stefano C D and Fontanella F, "Evolutionary Prototyping for Handwritten Recognition", *International Journal of Pattern Recognition and Artificial Intelligence,* vol 21, Number 1, pp. 157-178, 2007

[15] Freeman J A, Skapura D M, *Neural Networks: Algorithms, Applications and Programming Techniques*, Reading, MA: Adidison Wesley, 1991