Global Journal of Advanced Engineering Technologies and Sciences

DESIGN OF FININTE STATE MACHINES EMPLOYING BACK PROPAGATION IN ARTIFICIAL NEURAL NETWORKS

[1]Mohit Punjabi, [2]Ravi Kateeyare
[1]Research Scholar, [2]Asst. prof
Department of Electronics & Communication Engineering, CIIT Indore

ABSTRACT

Design of Finite State machines has a lot of applications ranging from the simple vending machines to complex robotics for space exploration. Recently the need for anticipative state machines exhibiting Artificial Intelligence has come up. In this paper, anticipative state machines have been designed using Artificial Neural Networks. Back propagation in ANN has been resorted to with the aim of attaining low errors in prediction and lesser training time, both of which are critical parameters for real time applications. It has been shown that the proposed technique renders lesser prediction errors with respect to increasing number of inputs as compared to previously existing techniques.

Keywords: Artificial Neural Network (ANN), Back Propagation, Levengerg-Marquardt (LM), Mean Square Error (MSE), Regression, Finite State Machines (FSM).

INTRODUCTION

Digital circuits have been the backbone of almost all electronic equipments working upon digital technology. Some common examples are:

1) A Simple Vending Machine
2) An ATM machine dispensing currency notes.
3) Complex electronic devices running interactive software and applications such as gaming consoles, mobile phones etc.

Digital Circuits which reach pre-defined finite states for some pre-defined user inputs are termed as interactive finite state machines. There are two ways in which a digital circuit can or finite state machine may be designed:

1) By knowing the internal circuitry of the system.
2) Knowing the exact input-output mapping of the system called the Truth Table.

Lately a new paradigm of digital circuits and finite state machines have evolved which work on real time applications and need to predict or forecast certain input-output relations based on previous input-output mapping of the system. Such systems find applications in:

1) Human Machine Interface (HMI) design
2) Interactive Gaming, for example playing chess with a personal computer.
3) Digital Machines working on hardware level cryptography etc.

Such systems need to predict behavioral patterns depending on previous patterns. Since the basic block of all such systems are digital circuits, hence it becomes mandatory to be able to predict the functionality of complex digital circuits using previous input-output mappings employing Artificial Intelligence (AI).

ARTIFICIAL INTELLIGENCE AND ARTIFICIAL NEURAL NETWORKS

Artificial intelligence can be defined as the development of computational systems competent enough to perform tasks which would otherwise require human intervention.
The fundamental requirement of understanding artificial intelligence is the perception of intelligence in the first place. Intelligence can be thought of as a sequential implementation of the following steps:

1) Accepting data.
2) Analyzing Data.
3) Figuring out regularities or patterns in the data.
4) Taking some decision based on the above steps.
5) Storing the experiences from every input and decision taken thereafter.

6) Making use of the stored experiences in taking future decisions

Any paradigm that is not natural, i.e. artificial and exhibits the above attributes is said to possess artificial intelligence.

One of the most common techniques of developing and implementing a system with artificial intelligence is the design of Artificial Neural Networks (ANN). The human brain is made of brain cells called neurons. Therefore the human brain is called a neural network. An artificial neural network is a computational machine which emulates the human neural network (brain).

Artificial Neural Network

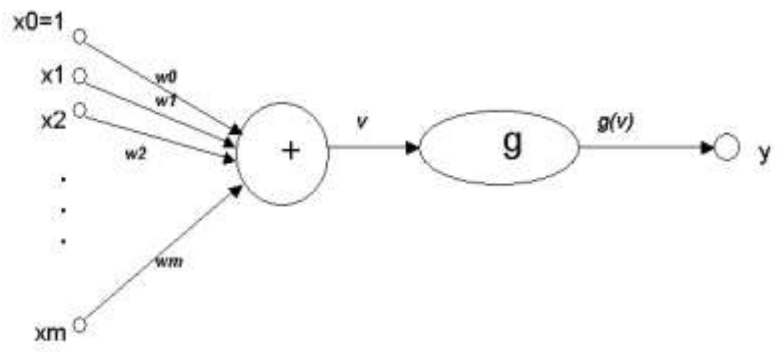

Fig 1: Artificial Neural Network

Here,
X represents the inputs
W represents the Weights or Experiences
g represents the bias

Artificial Neural Networks had developed as a major paradigm for Data Mining applications. Neural nets have gone through two major development periods -the early 60's and the mid 80's. They were a key development in the field of machine learning. Artificial Neural Networks were inspired by biological findings relating to the behavior of the brain as a network of units called neurons. The human brain is estimated to have around 10 billion neurons each connected on average to 10,000 other neurons. Each neuron receives signals through synapses that control the effects of the signal on the neuron. These synaptic connections are believed to play a key role in the behavior of the brain. The fundamental building block in an Artificial Neural Network is the mathematical model of a neuron as shown in Figure 1.6. The figure depicts the fact that

a) The artificial neural network takes data in a parallel manner just like the human brain.
b) Sums up all the inputs at an instant.
c) Analyzes is based on some mathematical function
d) Yields some output based on the previous steps.
e) Stores the experiences in the form of weights.

BACK PROPAGATION

Although the weights of a neural network tend to make subsequent error plummet, yet the rate at which the error reduces is critical in real time systems. Hence, it is prudent enough to design a system in which errors are fed back to the system in order to make it realize the amount of error present in every iteration. This technique is called back propagation. The commonly used techniques used for back propagation are

a) Levenberg-Marquardt (LM) algorithm
b) Bayesian Regularization (BR) algorithm
c) Scaled Conjugate Gradient (SCG)

We aim in designing both combinational and sequential circuits using ANN-Back propagation.

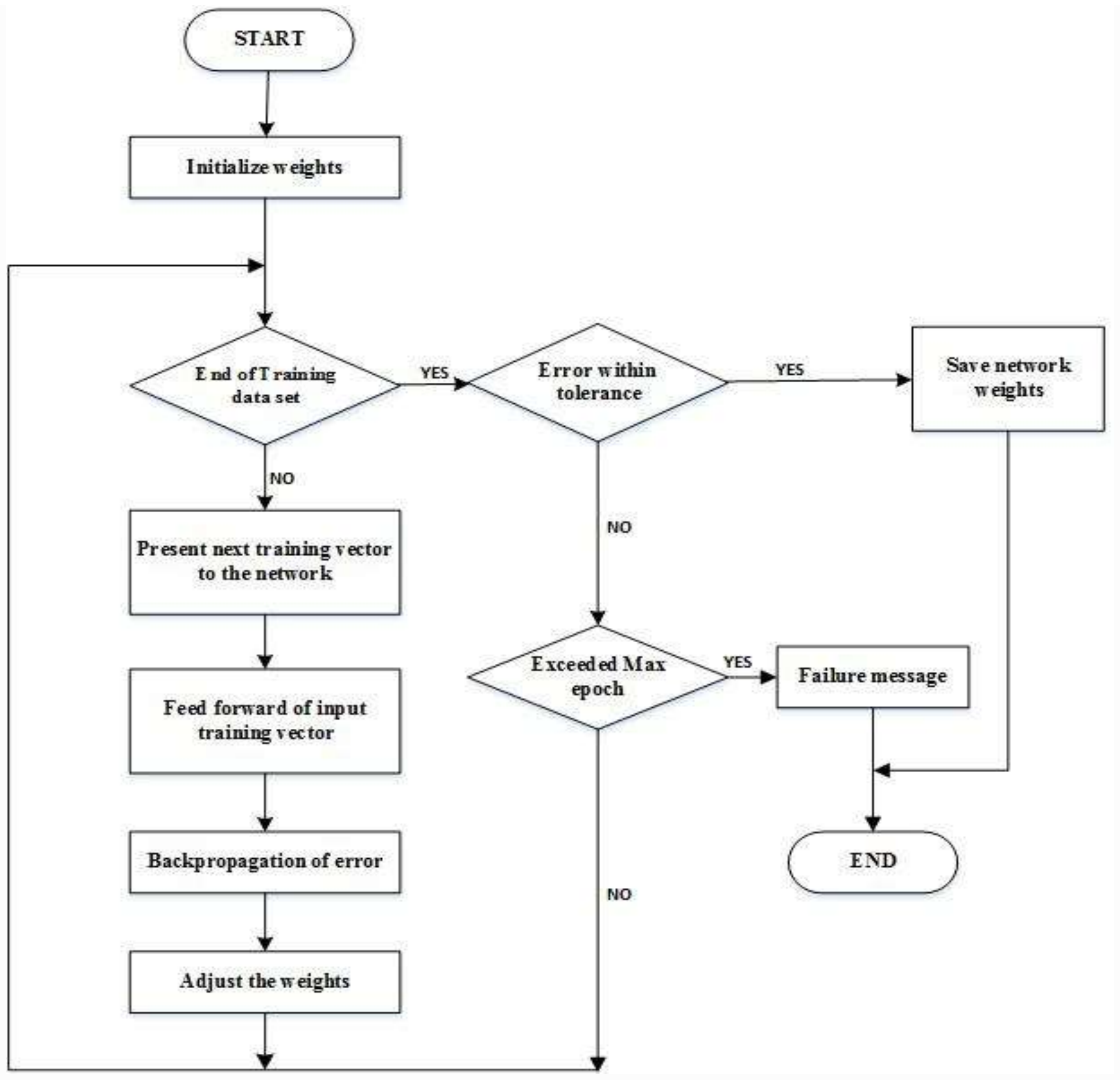

Fig.2: Block Diagram of Back propagation Neural Network.

PROPOSED METHODOLOGY

Reducing error function is the main reason for using this algorithm. Levenberg-Marquardt algorithm is a very efficient technique for minimizing a nonlinear function. It was first proposed by K. Levenberg in 1944 and then later modified by D. Marquardt in 1963 hence it was named on them. The algorithm contains many different variables like in present study we have output data (wind speed), weight in between neurons and error function, which govern the efficiency and success rate of network.

Levenberg-Marquardt algorithm is fast and has stable convergence. This algorithm was developed to approach 2ndorder training speed without calculating the Hessian matrix. When the performance function is in the form of a sum of squares, then the Hessian matrix and the gradient can be approached and calculated as,

$$H = J_k^T J_k \qquad (1)$$

$$g = J_k^T e \qquad (2)$$

Where J_k is the Jacobian matrix for kth input set, which contains first order derivatives of the network errors with respect to the weights and biases, e is representing network errors. Hence computing Jacobian matrix through a back propagation technique is far less complex than computing the Hessian matrix.
The Levenberg –Marquardt algorithm is actually a blend of the steepest descent method and the Gauss–Newton algorithm. The following is the relation for LM algorithm computation,

$$W_{k+1} = W_k - \left[J_K{}^T J_k + \mu I\right]^{-1} J_K{}^T e_k \qquad (3)$$

Where, I is the identity matrix, W_k is the current weight, W_{k+1} is the next weight and e_k is the last error, μ is combination coefficient.
It tries to combine the advantages of both the methods hence it inherits the speed of the Gauss–Newton method and the stability of the steepest descent method. The combination coefficient μ is multiplied by some factor (β) whenever a step would result in an increase in errore_{k+1} and when a step reduces $e_{k+1,}$ μ is divided by β. In this study, we used β=l0. When μ is large the algorithm converts to steepest descent while for small μ the algorithm converts to Gauss-Newton.

RESULTS

Consider the Finite State Machine (FSM) based Sequence Detector to Detect a Sequence **'1111'** i.e. to say the output of the circuit will be **LOGIC 1** when it receives an input **1111** and yields **LOGIC 0** for any other combination of the input.

Table.1 State Table of a Sequence Detector detecting the bit stream **1111**

S. No	STATE	OUTPUT
1	0000	0
2	0001	0
3	0101	0
4	0001	0
5	0010	0
6	**1111**	**1**
7	0101	0
8	0110	0
9	0100	0
10	**1111**	**1**

Now we analyze the performance of the proposed approach in accordance with different evaluation parameters:

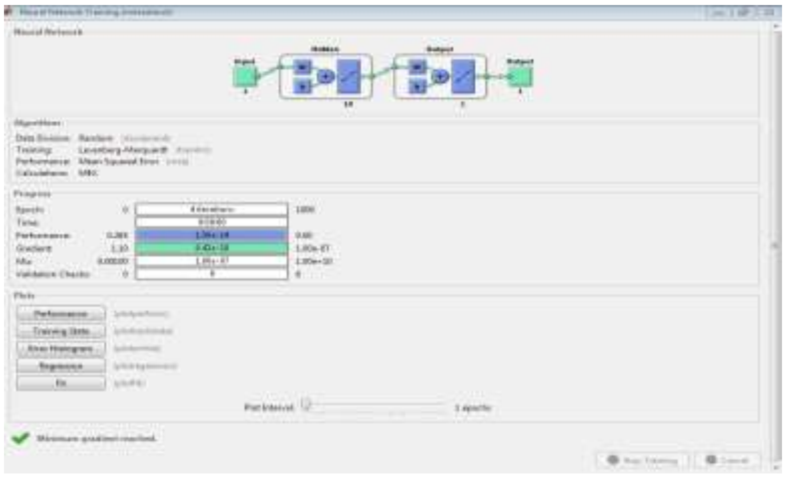
Data Division: Random
Training: Levenberg-Marquardt
Performance: Mean Squared Error
Calculations: MEX

Fig.3 MSE and Training Iterations for LM

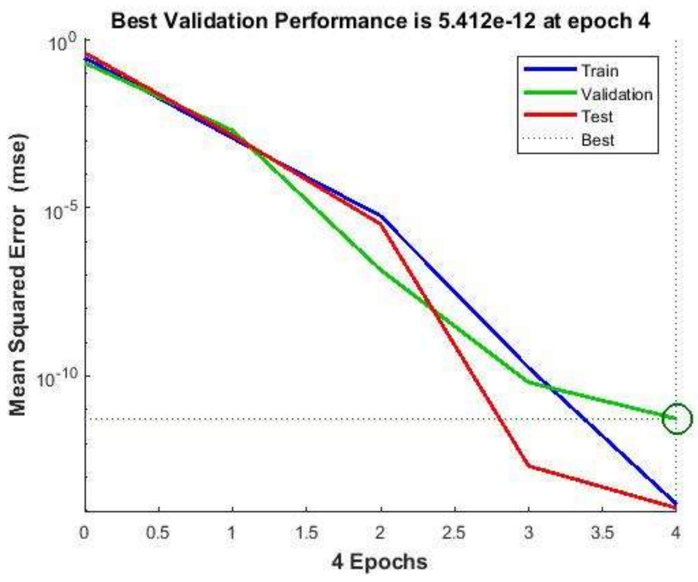

Fig.4 Training States using LM

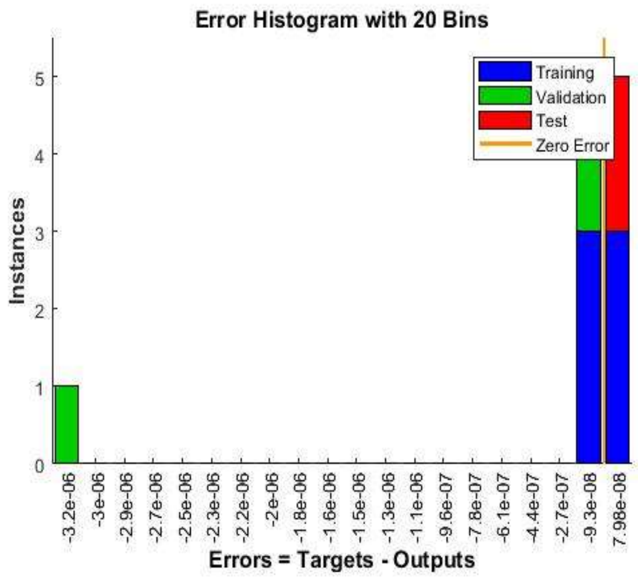

Fig.5 Error Histogram using LM

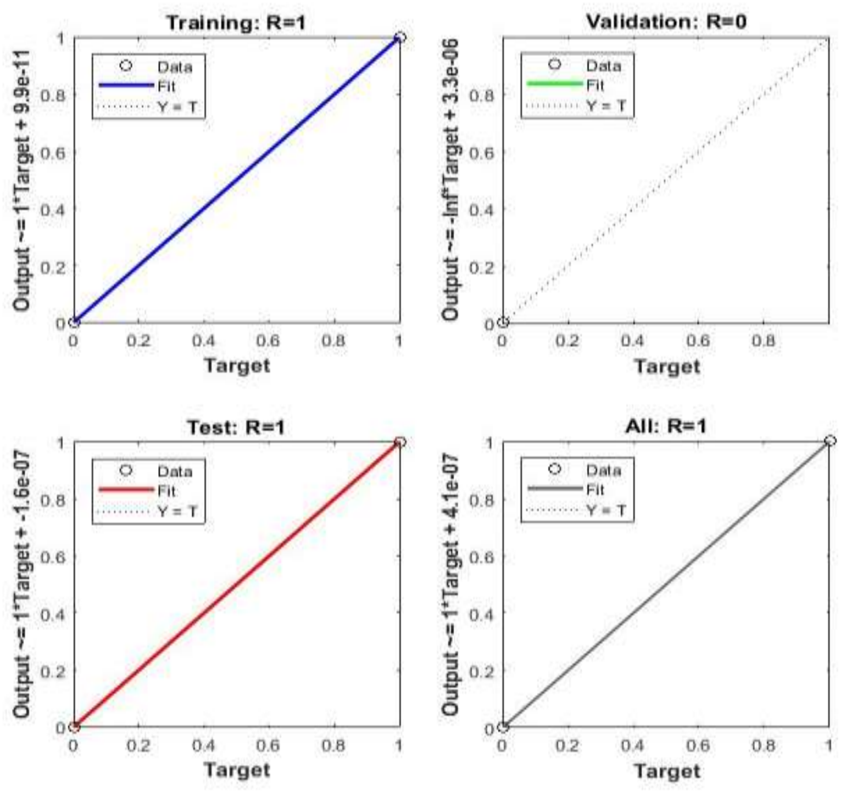

Fig.6 Regression Analysis using LM

Summary of results:

No. of Iterations:

LM: 04

MSE:

LM: 1.56 x 10^{-14}

REGRESSION

Regression is (approx 1).

CONCLUSION

It can be concluded from the previous discussions that the design and prediction of complex digital circuits is possible using Artificial Neural Networks (ANN). The benefit of such a system lies in the fact that there is no need of knowing the internal circuitry or the complete input output mapping of the digital system to predict its performance. The behavior of the circuit can be predicted by the ANN based system once the system has been trained. This approach finds extensive application in interactive gaming, Human Machine Interfaces (HMI), hardware level cryptography. It has been shown that the proposed approach attains very less magnitude of errors, relatively lesser number of iterations and a high value of regression (approx 1)

REFERENCES

1. On adaptive experiments for nondeterministic finite state machines Natalia Kushik · Khaled El-Fakih · Nina Yevtushenko · Ana R. Cavalli, Springer 2014
2. Deep Learning in Neural Networks: An Overview Technical Report IDSIA-03-14 / arXiv:1404.7828 v4 [cs.NE] (88 pages, 888 references) J¨urgenSchmidhuber The Swiss AI Lab IDSIA Istituto Dalle Molle di Studisull'Intelligenza Artificiale, IEEE 2014
3. State assignment for area minimization of sequential circuits based on cuckoo search optimization q Aiman H. El-Maleh, Sadiq M. Sait, AbubakarBala Elsevier 2015
4. Jordan Neural Network for Modelling and Predictive Control of Dynamic Systems Antoni Wysocki and Maciej Ławry´nczuk, IEEE 2015
5. VLSI Implementation of Deep Neural Network Using Integral Stochastic Computing Arash Ardakani, Student Member, IEEE, François Leduc-Primeau, Naoya Onizawa, Member, IEEE, Takahiro Hanyu, Senior Member, IEEE and Warren J. Gross, Senior Member, IEEE 2016
6. Predictive Abilities of Bayesian Regularization and Levenberg–Marquardt Algorithms in Artificial Neural Networks: A Comparative Empirical Study on Social Data Murat Kayri, MDPI, 2016
7. Weighting Finite-State Transductions with Neural Context Pushpendre Rastogi and Ryan Cotterell and Jason Eisner, NAACL-HLT Proceedings 2016
8. 'Design of logic gates using spiking neural P systems with homogeneous neurons and astrocytes-like control Tao Song, Pan Zhen, M.L. Dennis Wong, Xun Wang, Elsevier 2016
9. Memristor-Based Cellular Nonlinear/Neural Network: Design, Analysis, and Applications ShukaiDuan, Member, IEEE, Xiaofang Hu, Student Member, IEEE, Zhekang Dong, Student member, IEEE Transaction 2015
10. Low-Power Analysis of VLSI Circuit Using Efficient Techniques P.Rahul Reddy1, D.Prasad2, IJNTSE 2015
11. Recurrent Neural Networks: State Machines and Pushdown Automata C. Lee Giles, Alexander Ororbia II The Pennsylvania State University University Park, PA, USA.
12. Application of neural networks to efficient design of wireless and RF circuits and systems Ravender Goyal and Vladimir Vereme, AMSACTA 2000
13. Back propagation and Levenberg-Marquardt Algorithm for Training Finite Element Neural Network Arnold Reynaldi∗, Samuel Lukas†, Helena Margaretha∗, IEEE 2012
14. A Synthesis Flow for Sequential Reversible Circuits Mathias Soeken_ Robert Wille_ Christian Otterstedt_ Rolf Drechsler, IEEE 2014