ISSN: 2277-5528
Impact Factor: 4.015

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & MANAGEMENT

A SURVEY ON FINITE STATE MACHINE DESIGN USING ARTIFICIAL NEURAL NETWORKS

[1]Mohit Punjabi, [2]Ravi Kateeyare
[1]Research Scholar, [2]Asst. prof
[1,2]Department of Electronics & Communication Engineering, CIIT Indore
Punjabi.mohit79@gmail.com

ABSTRACT

Artificial Intelligence (AI) has taken the world by storm. Several applications which needed human intervention even in the near past are being seamlessly merged with AI based systems which are proving to be more competent and efficient. AI has made its presence felt in the field of digital electronics and stochastic computing as well. Presently applications based on the design of Finite State Machines (FSMs) are getting intertwined with AI based systems. This paper presents a survey and taxonomy on the implementation of AI using Artificial Neural Networks and how finite state machines can be designed using ANN structures.

Keywords: Finite State Machines (FSM), Artificial Intelligence (AI), Artificial Neural Network (ANN), Stochastic Computing.

INTRODUCTION

With the advent of digital technology arose the need for large computational machines and platforms. While conventional mechanisms use finite state machines for their implementation, yet it doesn't seem to suffice due to the following reasons:

a) Complexity of Finite State Machines for large data processing
b) Need of interactive and predictive nature of FSMs for applications such as gaming, prediction problems, automation, Human Machine Interfaces (HMI) etc.

Thus the field of stochastic computing or statistical computing needs systems to perform predicatively even if they are no explicitly programmed. The paradigm of Artificial Intelligence thus emerges as a necessity.

Intelligence can be crudely defined as the capability to follow the following steps:

a) Accept input data
b) Analyze Data
c) Recognize patterns or regularities in the data
d) Take a decision
e) Produce an output.

Humans exhibit the above attributes and so do other living organisms to different extents. Living organisms exhibit natural intelligence to different extents. In case we can design artificial systems to exhibit the above attributes, the Artificial Intelligence evolves. The most common way for it is to design an artificial Neural Network. Artificial Neural Networks (ANN) are computing systems or technique that are inspired by the learning architecture of human brain to discover the relations between the input and target variables of a system. Human brain consists of a large set of structural constituents, known as neurons, which form a well-connected network to respond to an input signal to perform all its computations / calculations in a certain complex task such as image and voice recognition task and they do this with incredible speed and accuracy. Neurons are simple processing units, which has the ability to store experimental data and which work as parallelly distributed processor. The speed of human brain is several thousand time faster than traditional computer because in brain unlike traditional computer as whole information is not passed from neuron to neuron they are rather encoded in the neuron network. This is reason why neural network is also named as connectionism.

INTRODUCTION TO ARTIFICIAL NEURAL NETWORKS

A biological model of neuron is basically comprised of dendrites, a cell body or soma, and an axon as shown in Figure 1. The cell body, also called the soma, holds the nucleus of neurons. The dendrites are the branches that are linked to the cell body and stretched in space around the cell body to receive signals from neighboring neurons. The axon works as a transmitter of the neuron. It sends signals to neighboring neurons. The synapse or synaptic terminal is the connection between the axon of one neuron and the dendrites of neighboring neutron, which is the communication linking between the two neurons.

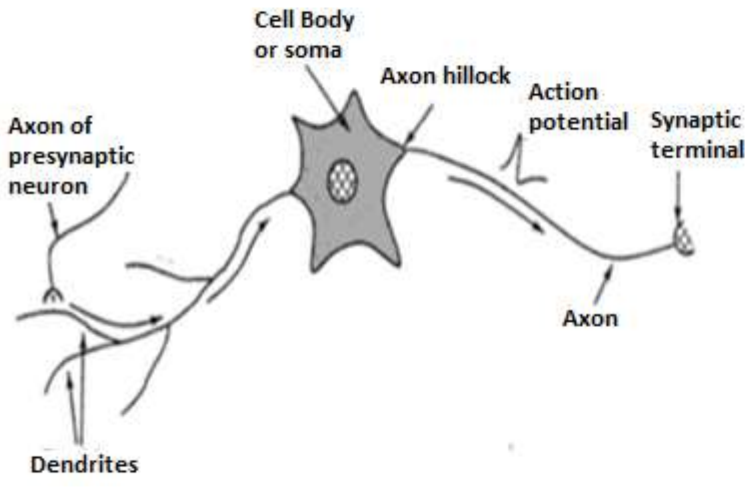

Fig.1 Biological Model of a Neuron.

The Electrochemical signals are communicated from the synapse. When the total signal received by a neuron is more than the synapse threshold, it causes the neuron to fire i.e., send an electrochemical signal to neighboring neurons. It is assumed that the change in the strength of the synaptic connection is the basis of human memory. This change is developed in ANN in the form of weights between neurons. In order to perform any type of action in our body, different parts of the body (sense organs) send signals which travel through other parts and reach the brain neuron's where the neuron processes it and generates the required output signal. It should be noted though that the output of a neuron may also be fed to another neuron. A collection of such neurons is called a neural network. The transformation of the biological model of neuron into a mathematical model is shown in the below figure 2."x" are different inputs which are weighted by a weight corresponding to a path that the signal travels. The neuron is then expected to add an effect in the form of an activation function ϴ and the complete signal then goes through a transformation S which produces the output of the neural network.

Consider a signals_1 travelling through a path p_1 from dendrites with weight $w_1$to the neuron. Then the value of signal reaching the neuron will be$s_1 . w_1$.If there are "n" such signals travelling through n different paths with weights ranging from w_1 to w_n and the neuron has an internal firing threshold value of θ_n , then the total activation function of the neuron is given by:

$$y = \sum_{i=1}^{n} X_i . W_i + \theta_i \qquad (1)$$

HereX_i represents the signals arriving through various paths, $W_i$represents the weight corresponding to the various paths and θ is the bias. The entire mathematical model of the neuron or the neural network can be visualized pictorially or the pictorial model can be mathematically modeled. The design of the neural network can be modeled mathematically and the more complex the neural design more is the complexity of the tasks that can be accomplished by the neural network. The above concept can be visualized by the following diagram:

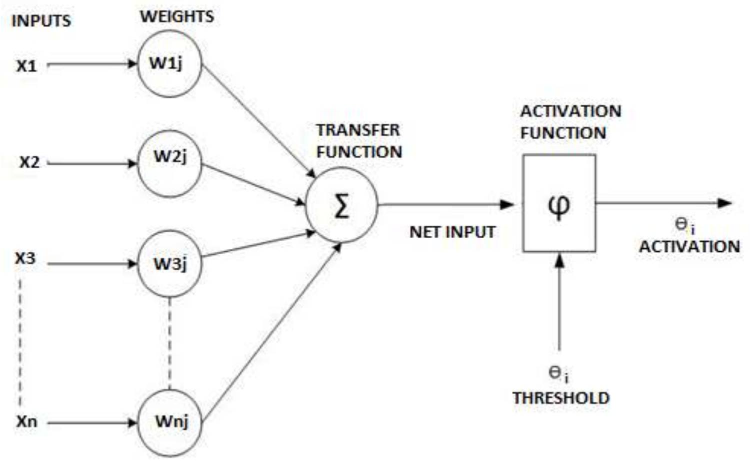

Fig.2 The mathematical formulation of the neural model.

The soul of the above model lies in the fact that the system so developed tries to mimic the working of human brain in terms of the following:

1) It works in a complex parallel computation manner
2) High speed of performance due to the parallel architecture.
3) It learning and adapt according to the modified link weights.

Work on ANN has been inspired right from its inception by the acknowledgement that the human brain computes in an entirely different way from the conventional digital computer.
ANN has an astonishing ability to find a relationship between completely non-linear data's which can be implemented successfully to detect trends and thus find the pattern followed by our targets which is impossible for human brains to notice.
ANN poses great ability to train itself based on the data provided to it for initial training. It has the tendency of self-organization during learning period and it can perform during real time operation.

ANN TOPOLOGIES

Single-layer feed forward Networks

In a layered neural network the neurons are organized in the form of layers. In the simplest form of a layered network, we have an input layer of source nodes that projects onto an output layer of neurons, but not vice versa. This network is strictly a feed forward type. In single-layer network, there is only one input and one output layer

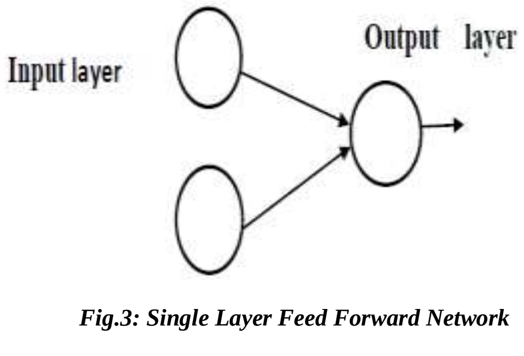

Fig.3: Single Layer Feed Forward Network

In feed forward networks, it's important to note that the flow of signal is from input nodes towards the output node but the signal cannot propagate backward from the output node towards the input layers i.e feed back is not permissible.

Multilayer feed forward Networks

The second class of a feed forward neural network has one or more hidden layers. The hidden layer acts as the functioning entity between the input and output layers. With increasing the number of hidden layers, the neural network can perform more complex tasks, although it may increase the complexity.

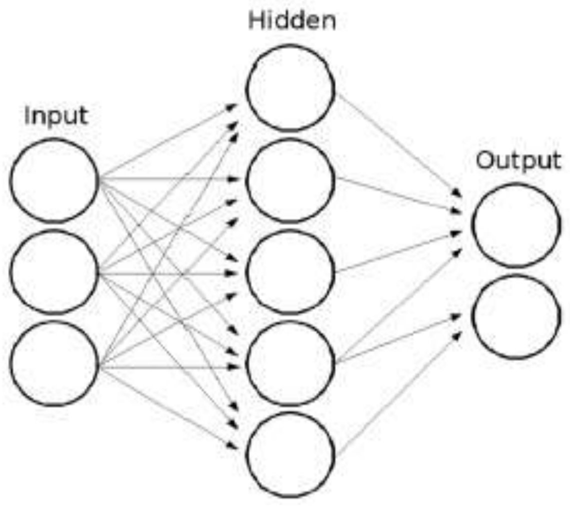

Fig.4: Multi layer feed forward network

Recurrent networks

A recurrent neural network has at least one feedback loop. Feedback refers to the technique in which some part of the output is applied back to the input. A recurrent network may consist of a single layer of neurons with each neuron feeding its output signal back to the inputs of all the other neurons. Self-feedback refers to a situation where the output of a neuron is fed back into its own input.

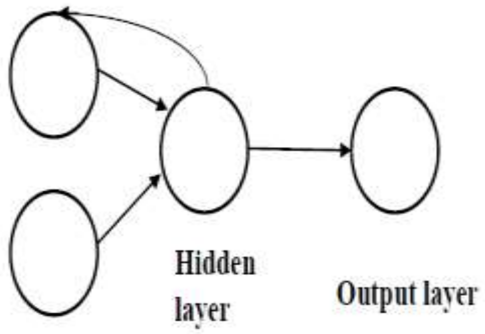

Fig.5: Recurrent Network

FINITE STATE MACHINES

There are many ways of modeling the behavior of systems, and the use of *state machines* is one of the oldest and best known. State machines allow us to think about the \state" of a system at a particular point in time and

characterize the behavior of the system based on that state. The use of this modeling technique is not limited to the development of software systems. In fact, the idea of state-based behavior can be traced back to the earliest considerations of physical matter.
From the discussion thus far, we can identify several key characteristics of a system that can be modeled with a finite state machine:

a) The system must be describable by a finite set of states.
b) The system must have a finite set of inputs and/or events that can trigger transitions between states.
c) The behavior of the system at a given point in time depends upon the current state and the input or event that occurs at that time.
d) For each state the system may be in, behavior is defined for each possible input or event.
e) The system has a particular initial state.

A simple state diagram of a finite state machine is given below

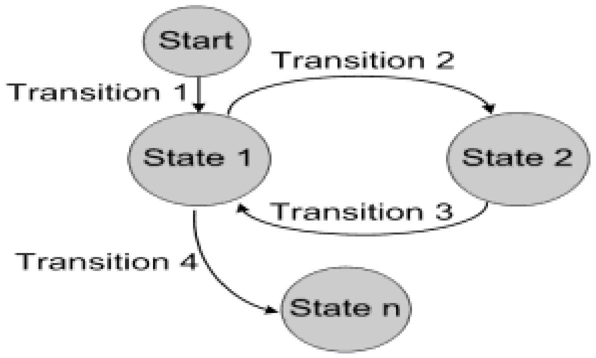

Fig.6: State Diagram of Finite State Machine

It can be seen that the finite state machine has finite states defined on the basis of current input and present input. If the state machine has finite states, then it’s called a finite state machine. It is often termed as a sequential machine. A ``sequential machine'' is a device in which the output depends in some systematic way on variables other than the immediate inputs to the device. These ``other variables'' are called the *state variables* for the machine, and depend on the history or state of the machine. For example, in a counter, the state variables are the values stored in the flip flops. The essence of a state table can be captured in a state diagram. A state diagram is a graph with labelled nodes and arcs; the nodes are the states, and the arcs are the possible transitions between states. Sequential State Machines or Finite State Machines have several applications such as Sequence Detectors in Vending Machines, ATM machines, Interactive Gaming just to name a few.

PREVIOUS WORK

In [1], NataliaKushik ,Khaled El-Fakih,Nina Yevtushenko and Ana R. Cavalli, in the paper “On adaptive experiments for nondeterministic finite state machines” proposed an algorithm for deriving adaptive homing and distinguishing experiments for non initialized nondeterministic finite state machines (NFSMs). Given a non-initialized complete FSM, a way for deriving adaptive homing/distinguishing experiments is proposed. Adaptive experiments are represented as special nondeterministic observable machines, called test cases. Necessary and sufficient conditions for having adaptive homing/ distinguishing test cases with minimal length for observable and non-observable nondeterministic FSMs are established.

In [2], J¨urgen Schmidhuber, in the paper "Deep Learning in Neural Networks: An Overview" proposed a study on Deep Learning in Neural Networks. Standard neural network (NN) consists of many simple, connected processors called neurons, each producing a sequence of real-valued activations. Input neurons get activated through sensors perceiving the environment and other neurons get activated through weighted connections from previously active neurons. By triggering actions some neurons may affect the environment. Learning or credit assignment is about finding weights that make the NN exhibit desired behaviour, such as driving a car. Depending on the specific problem and cognizance of how the neurons are connected, such behavior may require long causal chains of computational stages, where each stage transforms (often in a non-linear way) the aggregate activation of the network. Deep Learning is about accurately assigning credit across many such stages.

In [3], Aiman H. El-Maleh ,Sadiq M. Sait and AbubakarBala, in the paper "State assignment for area minimization of sequential circuits based on cuckoo search optimization" proposed the application of cuckoo search optimization (CSO) algorithm for solving the state assignment problem (SAP) of FSMs with the aim of minimizing area of the resulting sequential circuit. Results obtained from the CSO algorithm are compared with those obtained from binary particle swarm optimization (BPSO) algorithm, genetic algorithm (GA), and the well-known deterministic methods of NOVA and JEDI. The results indicate that CSO outperforms deterministic methods as well as other non-deterministic heuristic optimization methods. It presented the application of a recent optimization method, the cuckoo search optimization algorithm, to solve the state assignment problem in FSM. Experimental results on MCNC/LGSynth benchmark circuits show that CSO supersedes the rest and achieved better results than other non-deterministic heuristic optimization methods such as Genetic algorithm (GA) and binary particle swarm optimization (BPSO), and the well-known deterministic methods of NOVA and JEDI.

In [4],Antoni Wysocki and MaciejŁawry´nczuk, in the paper "Jordan Neural Network for Modelling and Predictive Control of Dynamic Systems" proposed and discussed the possibility of using a Jordan neural network as a model of dynamic systems and it highlights a Model Predictive Control (MPC) algorithm in which such a network is used for prediction. The Jordan network is a simple recurrent neural structure in which only one value of the process input signal (from the previous sampling instant) and only one value of the delayed output signal of the model (from the previous sampling instant) are used as the inputs of the network. It can be successfully used in predictive control.

In [5], Arash Ardakani, François Leduc-Primeau, Naoya Onizawa, Takahiro Hanyu and Warren J. Gross,in the paper "VLSI Implementation of Deep Neural Network Using Integral Stochastic Computing" suggested that an integer form of stochastic computation and introduce some elementary circuits. An efficient implementation of a DNN based on integral stochastic computing is put forward. An efficient stochastic implementation of a deep belief network is proposed using integral SC. Going by the simulation and implementation results ,it show that the proposed design reduces the area occupation by 66% and the latency by 84% with respect to the state of the art. It also indicates that the proposed design consumes 21% less energy than its binary radix counterpart. Moreover, the proposed architectures can save up to 33% energy consumption w.r.t. the binary radix implementation by using quasi-synchronous implementation without compromising on performance and keeping the efficiency intact.

In [6], Murat Kayri,in the paper "Predictive Abilities of Bayesian Regularization and Levenberg–Marquardt Algorithms in Artificial Neural Networks: A Comparative Empirical Study on Social Data" demonstrated the objective of the study which was to compare the predictive ability of Bayesian regularization with Levenberg–Marquardt Artificial Neural Networks. Since complex models are penalized in accordance with the Bayesian approach, this approach explores complex architecture smoothly. All in all, the model with two-neurons is the best architecture because of the highest importance level of predictors.

In [7], Pushpendre Rastogi, Ryan Cotterell and Jason Eisner, in the paper "Weighting Finite-State Transductions With Neural Context" proposed a new approach to deep learning to tasks such as morphological re inflection, which stochastically edits one string to get another .The traditional architecture is kept as such, and A stack of bidirectional

LSTMs reads the input string from left-to-right and right-to-left, in order to summarize the input context in which a transducer arc is applied. These learned characteristics and traits are intertwined with the transducer to define a probability distribution over aligned output strings, in the form of a weighted finite-state automaton. This lessens hand-engineering of features, allows learned features to examine unbounded context in the input string, and still permits exact inference through dynamic programming.

In [8], Tao Song et.al, in the paper "Design of logic gates using spiking neural P systems with homogeneous neurons and astrocytes-like control" proposed an approach to investigate the spiking neural P systems, i.e. SN P systems, with astrocyte-like control which were proven to have "Turing completeness" as computing models. In biological nervous systems, the operation of interacting neurons depends largely on the regulation from astrocytes. In this research, it is proposed and established SN P systems with astrocyte-like control to emulate logic AND, OR, NOT, NOR, XOR and NAND gates. The obtained SN P systems are simple and homogeneous, which infers that each neuron in the systems has the same unique spiking rule. The systems proposed in this work provide theoretical models to construct neural-like logic gates with universal information processing units. It will be beneficial if parallel hardware, such as GPU, can be utilized and harnessed to realize the neural-like digital logic gates and logic circuits.

EVALUATION PARAMETERS

The evaluation parameters for Finite State Machine Design employing ANN are described below:

The mean absolute error gives an idea about the average squared error and is defined as:

$$MAE \ = \frac{1}{N}\sum_{t=1}^{N} |A_t - \hat{A}_t| = \ \frac{1}{N}\sum_{t=1}^{N} |e_t| \quad (2)$$

The mean absolute percentage error (MAPE), is also used to measure prediction accuracy of a forecasting method. It expresses accuracy as a percentage, and is given by:

$$MAPE \ = \frac{100}{N}\sum_{t=1}^{N} \frac{|A_t - \hat{A}_t|}{A_t} \qquad (3)$$

The mean square error (MSE) is given by:

$$MSE \ = \ \frac{1}{N}\sum_{t=1}^{N} {e_t}^2 \qquad (4)$$

Here, N is the number of step points, A_t is the actual value and $\hat{A}_t$ is the forecasted value.

CONCLUSION

It can be concluded from the above discussions that the finite state machines or sequential machines have several real life applications such as Sequence Detectors in Vending Machines, ATM machines, Interactive Gaming, Cryptography, Decoding etc. While conventional design of finite state machines uses their internal architecture, modern real life applications that try to emulate human behaviour want the state transitions to change in real time. Hence the necessity of incorporating Artificial Intelligence in Finite State Machines comes into picture. This paper presents a survey on Artificial Neural Networks, their common topologies and finite state machines and its applications.

REFERENCES

I. On adaptive experiments for nondeterministic finite state machines Natalia Kushik · Khaled El-Fakih · Nina Yevtushenko · Ana R. Cavalli, Springer 2014

II. 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

III. State assignment for area minimization of sequential circuits based on cuckoo search optimization q Aiman H. El-Maleh, Sadiq M. Sait, AbubakarBala Elsevier 2015

IV. Jordan Neural Network for Modelling and Predictive Control of Dynamic Systems Antoni Wysocki and Maciej Ławry´nczuk, IEEE 2015

V. 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

VI. Predictive Abilities of Bayesian Regularization and Levenberg–Marquardt Algorithms in Artificial Neural Networks: A Comparative Empirical Study on Social Data Murat Kayri, MDPI, 2016

VII. Weighting Finite-State Transductions with Neural Context Pushpendre Rastogi and Ryan Cotterell and Jason Eisner, NAACL-HLT Proceedings 2016

VIII. '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

IX. 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

X. Low-Power Analysis of VLSI Circuit Using Efficient Techniques P.Rahul Reddy1, D.Prasad2, IJNTSE 2015

XI. Recurrent Neural Networks: State Machines and Pushdown Automata C. Lee Giles, Alexander Ororbia II The Pennsylvania State University University Park, PA, USA.

XII. Application of neural networks to efficient design of wireless and RF circuits and systems Ravender Goyal and Vladimir Vereme, AMSACTA 2000

XIII. Back propagation and Levenberg-Marquardt Algorithm for Training Finite Element Neural Network Arnold Reynaldi∗, Samuel Lukas†, Helena Margaretha∗, IEEE 2012

XIV. A Synthesis Flow for Sequential Reversible Circuits Mathias Soeken_ Robert Wille_ Christian Otterstedt_ Rolf Drechsler, IEEE 2014