# ARTIFICIAL NEURAL NETWORK TYPE LEARNING WITH SINGLE MULTIPLICATIVE SPIKING NEURON

Deepak Mishra[a], Abhishek Yadav[b], Sudipta Ray[c], Prem K. Kalra[a]

[a]Department of Electrical Engineering, IIT Kanpur, India
{dkmishra,Kalra}@iitk.ac.in

[b]Department of Electrical Engineering, GBPUAT Pantnagar, India
ayadav.iitk@gmail.com

[c]Indian Institute of Information Technology Allahabad, India
sray@iiita.ac.in

**ABSTRACT**
In this paper, learning algorithm for a single multiplicative spiking neuron (MSN) is proposed and tested for various applications where a multilayer perceptron (MLP) neural network is conventionally used. It is found that a single MSN is sufficient for the applications that require a number of neurons in different hidden layers of a conventional neural network. Several benchmark and real-life problems of classification and function-approximation are illustrated. It is observed that by incorporating nonlinear synaptic interaction, threshold variability, and spiking phenomena, learning in artificial neural networks can be made more efficient.

**Keywords:** multiplicative spiking neuron, learning, classification, function approximation.

## 1. INTRODUCTION

Many researchers have proposed several neuron models for artificial neural networks. Although all these models were primarily inspired from biological neuron, there is still a gap between philosophies used in neuron models for neuroscience studies and those used for artificial neural networks (ANN). Some of these models exhibit a close correspondence with their biological counterparts while others do not. Freeman (Freeman, 1988) has pointed out that while brains and neural networks share certain structural features, such as, massive parallelism, biological networks solve complex problems easily and creatively, but existing neural networks do not. He discussed the issues related to the similarities and dissimilarities between biological and artificial neural systems of present days. The main focus in the development of a neuron model for artificial neural networks is not only its ability to represent biological activities with its maximum intricacy, but also some mathematical properties e.g.; its capability as a universal function approximator. However, it can be advantageous for artificial neural networks if we can bridge the gap between biology and mathematics by investigating the learning capabilities of biological neuron models for the applications of classification, time-series prediction, function approximation etc. Here, we used the *spiking neuron model (SNM)* for this purpose.

History of artificial neural networks consists of the history of various neuron models and the history of learning methodologies. The first artificial neuron model was proposed by McCulloch and Pitts (McCulloch and Pitts, 1943) in 1943. They developed this neuron model based on the fact that the output of the neuron is unity if the weighted sum of its inputs is greater than a threshold value, otherwise the output is zero. Rosenblatt (Rosenblatt, 1958) proposed some modifications in the McCulloch-Pitts neuron model. Instead of a binary output, the Rosenblatt's neuron model creates a real-valued output representing the average firing rate of the cell. Durbin and Rumelhart (Durbin and Rumelhart, 1989) proposed the *sigma-pi* neuron model to incorporate the correlations among input pairs.

In 1949, Hebb (Hebb, 1949) proposed a learning rule that became initiative for ANNs. He postulated that the brain learns by changing its connectivity patterns. Widrow and Hoff (Widrow and Hoff, 1960) in 1960 presented the most analyzed and most applied learning rule known as least mean square rule. Later in 1985, Widrow and Sterns (Widrow and Sterns, 1985) found that this rule converges in the mean square to the solution that corresponds to least mean square output error if all input patterns are of same length. A single neuron of the above and many other neuron types proposed by several scientists and researchers are capable of linear classification (Sinha et. al., 2002). (Yadav et. al., 2003) incorporated various aggregation functions to model the nonlinear input-output relationships. In 2004, (Mishra et. al., 2004) investigated the chaotic behavior in neural networks that represent biological activities in terms of firing rate. (Scholles et. al., 1993) discussed biologically inspired artificial neurons and Feng and Li (Feg and Li, 2001) introduced neuronal models with current inputs. Training the integrate-and-fire model with the Informax principle was discussed in (Feng et. al., 2002) (Feng et. al., 2003).

Recently, spiking neural networks have been the subject of significant research reflecting the view that spikes have a key role in biological information processing (Gerstner, 1995) (Hopfield, 1995). New advances in neurophysiology have found that the difference in firing times could convey information about the input stimuli and that the relative order of firing times could be used as an alternative to rate coding (Gerstner and Kistler, 2002), (Mass, 1997). The first supervised training for this new computational approach was suggested in (Bohte et. al., 2000). However, in this model, a large number of parameters are to be adjusted. In our experiments, we found that the performance of the network is very much dependent upon the initial values of these parameters.

In the present work, a modified spiking neuron model is proposed and discussed. This modification accounts for nonlinear aggregation operation at dendrites. Moreover, probability of spike is used as an output instead of time of spike. We found that with such modifications the learning performance is drastically improved. In section 2, a brief discussion on the Integrate-and-Fire neuron model and Spiking neuron model is presented. Inspired from the relationship between timings of incoming spikes in correspondence to dynamics internal state variable a learning algorithm is proposed in Section 3. The comparison of the proposed model with classical multi layer perceptron (MLP) is described in section 4. In section 5, we concluded our work with a brief discussion.

## 2. BIOLOGICAL NEURONS

### 2.1. Architecture of a Biological Neuron

Networks of biological neurons compute with the help of fast traveling pulses called action potentials. A neuron is the fundamental building block of biological neural networks. A typical neuron has three major parts: soma, axon and dendrites. Dendrites form a dendritic tree which is a very fine bush of thin fibers around the neuron's body. Dendrites receive information from neurons through axons i.e., long fibers that serve as transmission lines. An axon is a long cylindrical connection that carries impulses from the neuron. The end part of an axon splits into a fine arborization which terminates in a small endbulb almost touching the dendrites of neighboring neurons. The axon-dendrite contact organ is called synapse. Details of the biological neuron can be found in (Koch, 1999) (Tuckwell, 1988). Modeling such a complex system is difficult task and neuronal dynamics model often has to be drastically simplified in order to make both the computation and analysis somewhat tractable. In following subsection we will discuss two such models that mimic the biological activities of single neuron.

### 2.2. Integrate-and-Fire Neuron Models

The integrate-and-Fire (IF) neuron is the simplest of the threshold-fire neuron models. It is based on the membrane potential equations of HH model (Hodgkin and Huxley, 1952) with the membrane potential currents omitted. Although integrate and-fire model is a very simple model, it captures almost all of the important properties of the cortical neuron. Figure 1 shows the basic circuit of this model consists of a capacitor $C$ in parallel with a resistor $R$ driven by a current $I_{EXT}$ . The driving current can be split into two components, $I_{EXT} = I_R(t) + I_C(t)$. The first component is the resistive current which passes through the linear resistor $R$ and the second component charges the capacitor $C$.

$$I_{EXT} = \frac{v}{R} + C\frac{dv}{dt} \tag{1}$$

where, $v(t)$ is the membrane potential. A spike occurs when $v(t)$ reaches a threshold $V_{TH}$. After the occurrence of a spike, next spike cannot occur during refractory period $T_{REF}$ .

This model divides the dynamics of the neuron into two regimes: the subthreshold and the suprathreshold. The Hodgkin-Huxley equations (Hodgkin and Huxley, 1952) show that in subthreshold regime, sodium and potassium active channels are almost close. Therefore, the corresponding terms can be neglected in the voltage equation of the Hodgkin-Huxley model. This gives a first order linear differential equation similar to equation 1. In case of the suprathreshold region, if the voltage hits the threshold at time $t_0$, a spike at time $t_0$ will be registered and the membrane potential will be reset to $V_{RESET}$ . The system will remain there for a refractory period $T_{REF}$ . Figure 2 shows the response of an integrate-and-fire model. The solution of the first-order differential equation (equation 2) describing the dynamics of this model in sub-threshold region can be found analytically. With $v(0) = V_{REST}$, the solution of equation 2 is given by equation 3.

$$C\frac{dv}{dt} = G_L(V_L - v) + I_{EXT} \tag{2}$$

Here $G_L$ is leakage conductance.

$$v(t) = \frac{I_{EXT}}{G_L}\left(1 - e^{-tG_L/C}\right) + \left(V_{REST} - V_L\right)e^{-tG_L/C} + V_L \tag{3}$$

Let us assume that $v(t)$ hits $V_{TH}$ at $t = T_{TH}$. Thus

$$V_{TH} = \frac{I_{EXT}}{G_L}\left(1 - e^{-T_{TH}G_L/C}\right) + \left(V_{REST} - V_L\right)e^{-T_{TH}G_L/C} + V_L \tag{4}$$

Therefore, $T_{TH}$ can be written as

$$T_{TH} = \frac{C}{G_L}\ln\left(\frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L)}\right) \tag{5}$$

Interspike interval $T_{ISI}$ is the summation of $T_{TH}$ and $T_{REF}$ . Thus

$$T_{ISI} = T_{TH} + T_{REF} \tag{6}$$

Therefore,

$$T_{ISI} = \frac{C}{G_L}\ln\left(\frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L}\right) + T_{REF} \tag{7}$$

Frequency $f$ is reciprocal of interspike interval and hence is given by

$$f = 1\Bigg/\left(\frac{C}{G_L}\ln\left(\frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L}\right) + T_{REF}\right) \tag{8}$$

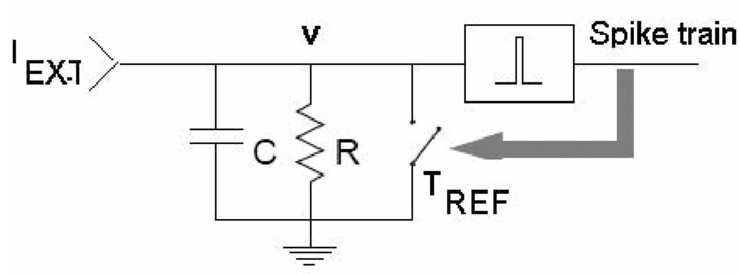


Figure 1. Circuit diagram of an integrate-and-fire neuron model

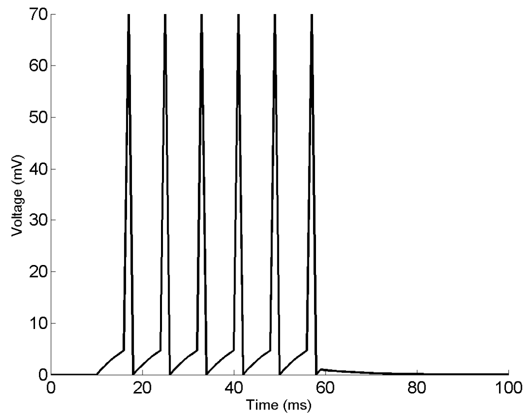


Figure 2. Response of an integrate-and-fire neuron model

**2.3. Spiking Neuron Models**

A spiking neuron is represented by voltage across its cell membrane and a threshold (Gerstner and Kistler, 2002). The status of a neuron is determined by integration of its excitatory and inhibitory post synaptic potentials (PSP). When its membrane potential reaches a certain threshold, the neuron generates a spike which is propagated to other neurons. The synapse is responsible for transforming the spike into a PSP. A typical shape of a PSP is shown in Figure 3. A spike takes certain time, called synaptic delay, to reach the post synaptic neuron. We assume that a neuron has n number of immediate predecessors called pre-synaptic neurons and receives a set of spikes with firing times $t_{i,}$, $i = 1, 2, ..n$. At most one spike is generated by each neuron during the simulation interval (the presentation of an input pattern), and fires when internal state variable reaches a threshold. Dynamics of the internal state variable $x(t)$ is determined by the impinging spikes, whose impact is described by the spike-response function $\varepsilon(t)$ weighted by the synaptic efficacy or weight $w_i$ (Bohte et. al. 2000).

$$x(t) = \sum_{i=1}^{n} w_i \,\varepsilon(t - t_i) \tag{9}$$

$w_i$ is positive for excitatory synapse and negative for inhibitory synapse. The spike-response function $\varepsilon(t - t_i)$, which is referred as PSP, models how the arrival of a single spike changes the membrane potential of the target neuron as a function of time-since-impact. The height of PSP is modulated by synaptic weight $w_i$ to obtain the effective PSP at the target neuron due to a spike from neuron-*i*. The commonly used spike response function is given in (Bohte et. al. 2000) and can be expressed by equation 10.

$$\varepsilon(t) = \frac{t}{\tau} e^{1-\frac{t}{\tau}} \tag{10}$$

As extension of this model, we can consider synaptic delay $\delta_i$ associated with each input. Moreover, time-constant $\tau$ can be considered different for different inputs. Thus

$$\begin{aligned} x(t) &= \sum_{i=1}^{n} w_i \varepsilon\left(t - t_i - \delta_i\right) \\ &= \sum_{i=1}^{n} w_i \left(\frac{t - t_i - \delta_i}{\tau_i}\right) e^{1-((t-t_i-\delta_i)/\tau_i)} \end{aligned} \tag{11}$$

Thus, the state variable $x$, at a specified time instant $t_0$ is

$$x_0 = \sum_{i=1}^{n} \left[\left(a_i + b_i t_i\right) e^{\left(c_i + d_i t_i\right)}\right] \tag{12}$$

Where

$$a_i = w_i \left(\frac{t_0 - \delta_i}{\tau_i}\right)$$

$$b_i = \frac{-w_i}{\tau_i}$$

$$c_i = 1 - \left(\frac{t_0 - \delta_i}{\tau_i}\right)$$

$$d_i = 1 + \frac{w_i}{\tau_i}$$

These two neuron model presents the biophysical phenomenon in terms of mathematical equations. In our previous work (Yadav et. al., 2005), we have used interspike interval relationship (equation 7) of integrate and fire neuron model for the learning purpose. In our present work we mainly concentrate on the timing of spikes in Spiking neuron model in order to derive an aggregation function for the learning of a neural network. The details of the proposed model are explained in following sections.

## 3. THE PROPOSED MULTIPLICATIVE SPIKING NEURON MODEL

Inspired from the relationship between timings of incoming spikes $t_i$ and state variable $x$ (equations 11 and 12) for spiking neuron model, following aggregation function is assumed instead of the weighted sum of a conventional neuron multiplicative like operation is considered:

$$x_{net} = \left[\prod_{i=1}^{n} \left(a_i + b_i x_i\right)\right] \times \left[e^{\sum_{i=1}^{n}\left(c_i + d_i x_i\right)}\right] \tag{13}$$

where, $n$ is number of inputs. $x_i$ is analogous to the input spike time $t_i$ and $x_{net}$ is analogous to the state variable $x$ at a specified time $t$. It is to be noted that parameters $ai$, $b_i$, $c_i$ and $d_i$ have been used to represent the constant values that depend upon $\tau_i$, $\delta_i$, $w_i$ and the specified time $t$ at which value of the state variable is to be calculated.

In view of evidence in support of the presence of multiplicative-like operations in the nervous system (Koch and Poggio, 1992), multiplication of net inputs to the activation function is considered. Instead of time of spike in the post synaptic neuron, we considered the spiking probability y till a specified time t as output. $y$ is a nonlinear function of $x_{net}$ and should have the following properties:

- It is a monotonically increasing function as probability of spike is increased for increasing $x_{net}$.
- It is almost zero for low values of $x_{net}$ as there is negligible probability of spike if $x_{net}$ is small. This is true even if threshold variability is taken into account.

- It is almost unity for high values of $x_{net}$ as spike is almost certain for large values of $x_{net}$. This is also true even if threshold variability is taken into account.

Assumption of constant threshold would lead to a special case of this function. In that case, the probability of spike would either be zero or one. Incorporation of threshold variability in our model is achieved by assuming this function as the sigmoid function. Thus

$$y = \frac{1}{1+e^{-x_{net}}} \tag{14}$$

### 3.1. Biological Significance of MSN

The MSN model is inspired from the fact that the actual shape of action potential does not contain any neuronal information. It is the timing of spikes that matters. A substantial body of evidence supports the presence of multiplicative-like operations in nervous system (Koch and Poggio, 1992). Physiological and behavioral data strongly suggest that the optomotor response of insects to moving stimuli is mediated by a correlation-like operation (Koch, 1999). Another instance of a multiplication-like operation in the nervous system is the modulation of receptive field location of neurons in the posterior parietal cortex by the eye and head positions of monkey (Koch, 1999). Multiplication operation is used for aggregation of inputs to the artificial neuron in many research papers including (Yadav et. al., 2003). In this work, we incorporate this multiplication operation while aggregating inputs to the activation function.

### 3.1. Development of the Training Algorithm

A simple steepest descent method is applied to minimize the following error function:

$$e = \frac{1}{2}(y-t)^2 \tag{15}$$

where, $t$ is target (not time) and $y$ is actual output of neuron. $e$ is function of parameters $a_i, b_i$, $c_i$ and $d_i$, $i = 1, 2, ...n$. Therefore, the parameter update rule (weight update rule) can be expressed in terms of following equations:

$$a_i^{new} = a_i^{old} - \eta \frac{\partial e}{\partial a_i} \tag{16}$$

$$b_i^{new} = b_i^{old} - \eta \frac{\partial e}{\partial b_i} \tag{17}$$

$$c_i^{new} = c_i^{old} - \eta \frac{\partial e}{\partial c_i} \tag{18}$$

$$d_i^{new} = d_i^{old} - \eta \frac{\partial e}{\partial d_i} \tag{19}$$

For $i = 1, 2, 3, ...n$. Partial derivatives of e with respect to parameters $a_i$, $b_i$, $c_i$ and $d_i$, $i = 1, 2, ...n$ can be given by following equations:

$$\frac{\partial e}{\partial a_i} = (t-y)y(1-y)\frac{x_{net}}{a_i + b_i x_i} \tag{20}$$

$$\frac{\partial e}{\partial b_i} = (t-y)y(1-y)\frac{x_{net} x_i}{a_i + b_i x_i} \tag{21}$$

$$\frac{\partial e}{\partial c_i} = (t-y)y(1-y)x_{net} \tag{22}$$

$$\frac{\partial e}{\partial d_i} = (t-y)y(1-y)x_{net} x_i \tag{23}$$

# 4. ILLUSTRATIVE EXAMPLES
## 4.1. Classification Problems
### 4.1.1 XOR Problem:

The XOR problem, as compared with other logic operations (NAND, NOR, AND and OR), is probably one of the best and most used nonlinearly separable pattern associator and consequently provides one of the most common examples of artificial neural systems for input remapping. We compared the performance of multiplicative spiking neuron (MSN) with that of multilayer perceptron (MLP). For this purpose, we considered an MLP with 3 hidden units. Figure 4 shows the training curves for MLP and MSN. It is clear that the proposed model takes only 52 iterations while MLP takes 523 iterations for training to achieve an MSE of order of 0.0001. Table I exhibits the comparison between MLP and MSN in terms of the deviation of actual outputs from corresponding targets. It can be seen here that the performance of MSN is almost same as that of MLP. It means that a single MSN is capable to learn XOR relationship 10 times faster than an MLP with 3 hidden units. Table 2 shows the comparison of training and testing performance with MLP and MSN for XOR-problem.

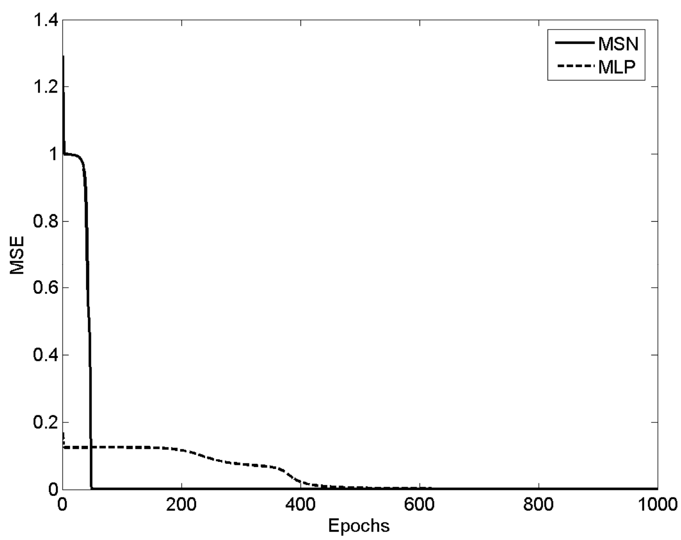


Figure 4. Training curves with MLP and MSN for XOR-problem. Training with MSN is significantly faster than that with MLP. For same mean square error (MSE), MLP takes 10 times more number of epochs than MSN.

**Table 1. Outputs of MSN and MLP for XOR problem**

| Input | Target | Output with MLP having 4 units | Output with MSN having one unit |
|---|---|---|---|
| 0 0 | 0 | 0.0001 | 0.0141 |
| 0 1 | 1 | 0.9890 | 0.9926 |
| 1 0 | 1 | 0.9922 | 0.9858 |
| 1 1 | 0 | 0.0204 | 0.0005 |

**Table 2. Comparison of training and testing performance for XOR problems**

| S. No. | Parameter | MLP | MSN |
|---|---|---|---|
| **1** | Training Goal, in terms of MSE | 0.00015 | 0.00015 |
| **2** | Iterations Needed | 523 | 52 |
| **3** | Training Time in seconds | 0.54 | 0.12 |
| **4** | Testing Time in seconds | 0.01 | 0.04 |
| **5** | MSE for Testing Data | 0.0001495 | 0.0001139 |
| **6** | Correlation Coefficient | 0.99989 | 0.99994 |
| **7** | Percentage Misclassification | 0 % | 0 % |
| **8** | Number of Parameters | 11 | 8 |

**4.1.2 Three-bit parity Problem:**

The 3-input XOR has been a very popular benchmark classification problem among the researchers of ANN. The problem deals with the mapping of 3-bit wide binary numbers into its parity. If the input pattern consists the odd numbers of 10s then the parity is 1, otherwise 0. This is a difficult problem because the patterns that are close in the sample space, i.e. the numbers that differ in only one bit, require their classes to be different. For comparison of the performance with MSN and MLP in case of three-bit Parity Problem, we considered MLP with 5 hidden units. Figure 5 shows training curves with conventional multilayer perceptron and the proposed single MSN model while training the artificial neural systems for three-bit *Parity Problem*. It is found that the proposed model takes only 25 iterations as compared to 1823 iterations taken by MLP for training to achieve an MSE of the order of 0.0001. Table 3 exhibits the comparison between MLP and MSN in terms of the deviation of actual outputs from corresponding targets. It can be observed that the performance of MSN is similar to MLP but MSN is capable to learn this relationship almost 30 times faster than that in case of an MLP with 5 hidden neurons. Table 4 shows the comparison of training and testing results with MLP and MSN while solving the *three-bit Parity problem*.

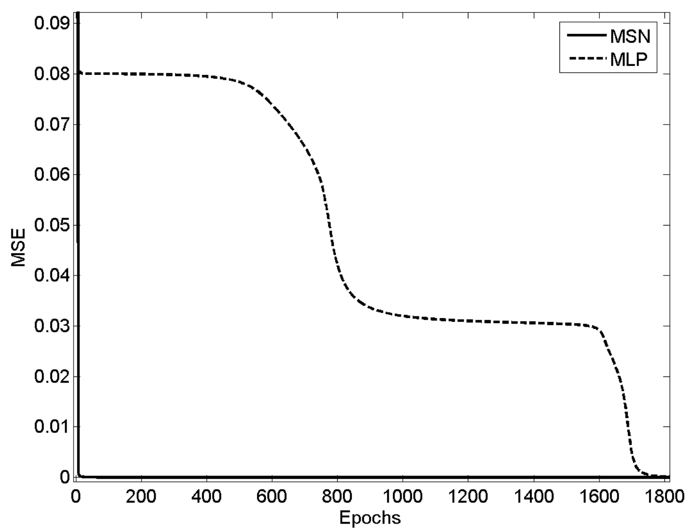


Figure 5. Training curves with MLP and MSN for three-bit parity problem. Training with MSN is significantly faster than that with MLP. For same mean square error (MSE), MLP takes 70 times more number of epochs than MSN.

Table 3. Outputs of MSN an MLP for three bit parity problem

| Input | Target | Output with MLP having 5 unit | Output with MSN having one unit |
|---|---|---|---|
| 0 0 0 | 0 | 0.0450 | 0.0377 |
| 0 0 1 | 1 | 0.9992 | 0.9899 |
| 0 1 0 | 1 | 0.9956 | 0.9895 |
| 0 1 1 | 0 | 0.0024 | 0.0067 |
| 1 0 0 | 1 | 1.0000 | 0.9856 |
| 1 0 1 | 0 | 0.0322 | 0.0164 |
| 1 1 0 | 0 | 0.0002 | 0.0064 |
| 1 1 1 | 1 | 0.9984 | 0.9658 |

Table 4. Comparison Of Training And Testing Performance For 3-Bit Parity Problem

| S. No. | Parameter | MLP | MSN |
|---|---|---|---|
| 1 | Training Goal, in terms of MSE | 0.0004 | 0.0004 |
| 2 | Iterations Needed | 1823 | 25 |
| 3 | Training Time in seconds | 1.65 | 0.17 |
| 4 | Testing Time in seconds | 0.07 | 0.05 |
| 5 | MSE for Testing Data | 0.0004 | 0.0004 |
| 6 | Correlation Coefficient | 0.9996 | 0.9997 |
| 7 | Percentage Misclassification | 0 % | 0 % |
| 8 | Number of Parameters | 22 | 12 |

**4.1.3 Iris data:**

This classic data of Anderson and Fisher pertains to a four-input, three class classification problem. The first four columns indicate petal and sepal widths and lengths of various iris flowers, and the fifth column indicates the appropriate class (Setosa, Versicolor, and Virginia). For comparison of performance with MSN and MLP in case of Iris data problem, we considered MLP with 5 hidden units. Figure 6 shows the comparison of training curves of conventional multilayer perceptron and the proposed single MSN model. It is clear that the proposed model achieves an MSE of 0.005 in 500 iterations while MLP achieves an MSE of 0.01 in same number of iterations. It is observed that the performance of MSN is almost same as compared with MLP but MSN with less number of parameters is capable to learn this relationship faster than an MLP having 5 hidden neurons. Moreover, we found less number of misclassified points with the proposed model. Table 5 shows the comparison of training and testing performance with MLP and MSN while solving the Iris data problem.

Table 5. Comparison of training and testing performance for Iris data set

| S. No. | Parameter | MLP | MSN |
|---|---|---|---|
| **1** | Training Goal, in terms of MSE | 0.01 | 0.005 |
| **2** | Iterations Needed | 500 | 500 |
| **3** | Training Time in seconds | 0.68 | 0.31 |
| **4** | Testing Time in seconds | 0.04 | 0.04 |
| **5** | MSE for Testing Data | 0.0155 | 0.0059 |
| **6** | Correlation Coefficient | 0.9977 | 0.9969 |
| **7** | Percentage Misclassification | 6 % | 4 % |
| **8** | Number of Parameters | 22 | 16 |

**4.2. Function Approximation Problems**

**4.2.1 Internet Traffic Data:**

Short term internet traffic data was supplied by HCL Infinet Ltd. (a leading Indian ISP). This data represents weekly internet traffic (in kbps) with a 30-minute average. Four measurements $y(t-1)$, $y(t-2)$, $y(t-4)$ and $y(t-8)$ were used to predict $y(t)$. For comparison of the performance with MSN and MLP with Internet Traffic Data, we considered MLP with 6 hidden neurons. Figure 7 shows the comparison of MLP and MSN for Internet Traffic Data. It is found that the proposed model exhibits faster training on this data. Figure 8 shows the comparison between MLP and MSN in terms of the deviation of actual outputs from corresponding targets. 200 data samples were used for training and rest of the data were used for testing. It can be observed that the performance of MSN for training data is similar as that of MLP while its performance is better for testing data. Table 6 shows the comparison of training and testing performance with MLP and MSN for the Internet Traffic Data.

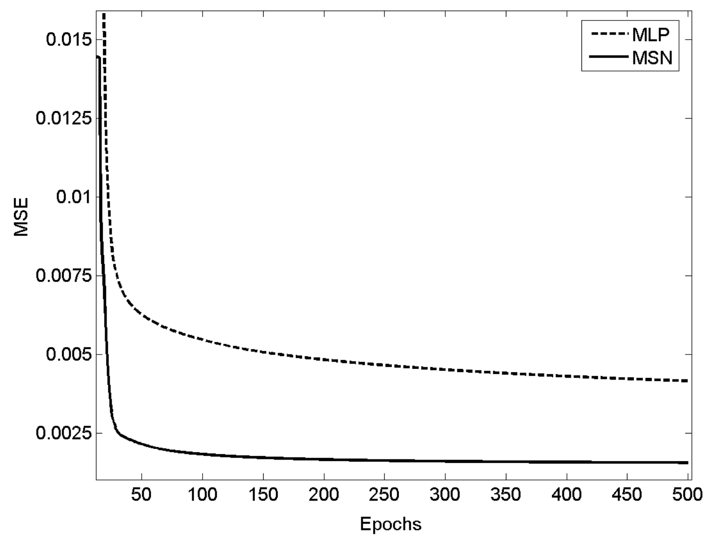


Figure 7. Training curve for Internet Traffic Data

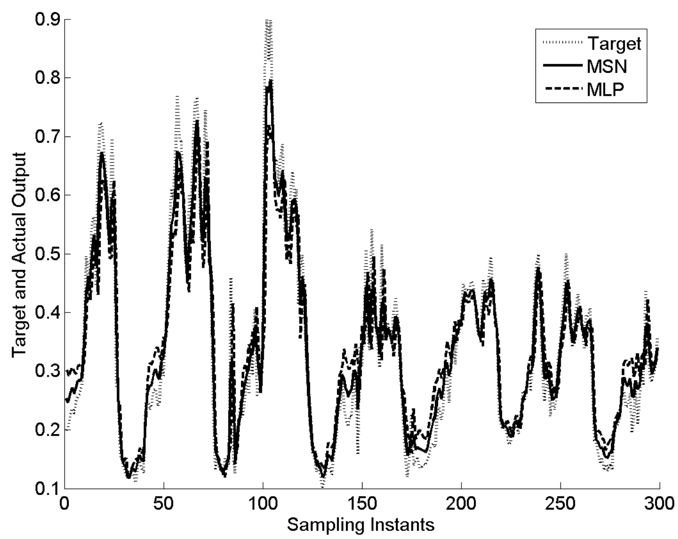


Figure 8. Target and Actual Output with MLP and MSN, for Internet-Traffic Data

Table 6. Comparison of training and testing performance for Internet traffic data

| S. No. | Parameter | MLP | MSN |
|---|---|---|---|
| 1 | Training Goal, in terms of MSE | 0.005 | 0.002 |
| 2 | Iterations Needed | 500 | 500 |
| 3 | Training Time in seconds | 0.78 | 0.22 |
| 4 | Testing Time in seconds | 0.08 | 0.06 |
| 5 | MSE for Testing Data | 0.0074 | 0.0033 |
| 6 | Correlation Coefficient | 0.9974 | 0.9996 |
| 7 | Number of Parameters | 22 | 16 |

### 4.2.2 Electroencephalogram Data:

Electroencephalogram (EEG) data used here was taken from website of Colostate university. Presence of randomness and chaos (Mishra et. al., 2004) in this data makes it interesting for neural network related research. In this problem, four measurements $y(t-1)$, $y(t-2)$, $y(t-4)$ and $y(t-8)$ were used to predict y(t). Figure 9 shows the comparison of MSE versus number of epochs curves with conventional multilayer perceptron and the proposed single MSN model while training the artificial neural systems for EEG data. It is clear that the proposed model exhibits faster training. Figure 10 shows the comparison between MLP and MSN in terms of the deviation of actual outputs from corresponding targets. 100 samples were used for training. It can be seen that the performance of MSN for training as well as testing data is much better than that of MLP. It means that a single MSN is capable to learn this relationship faster than that in case of an MLP with 6 hidden neurons and its performance on seen as well as unseen data is significantly better. Table 7 shows the comparison of training and testing performance with MLP and MSN while applying for the EEG Data.

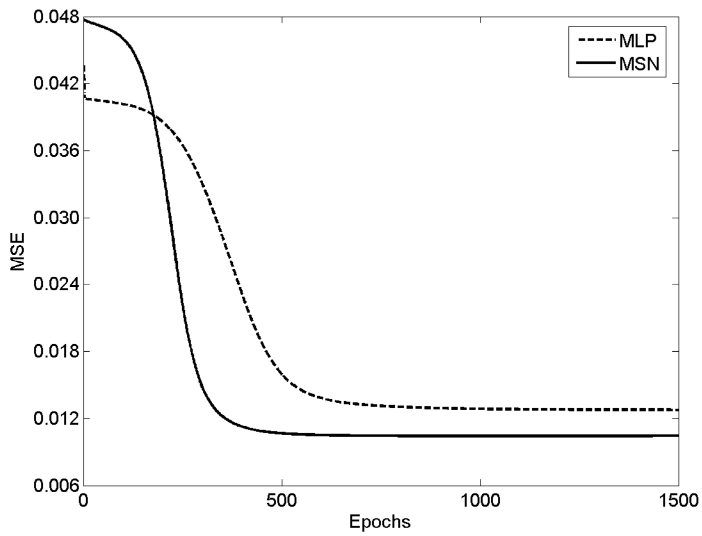


Figure 9. Training curve for *EEG Data*

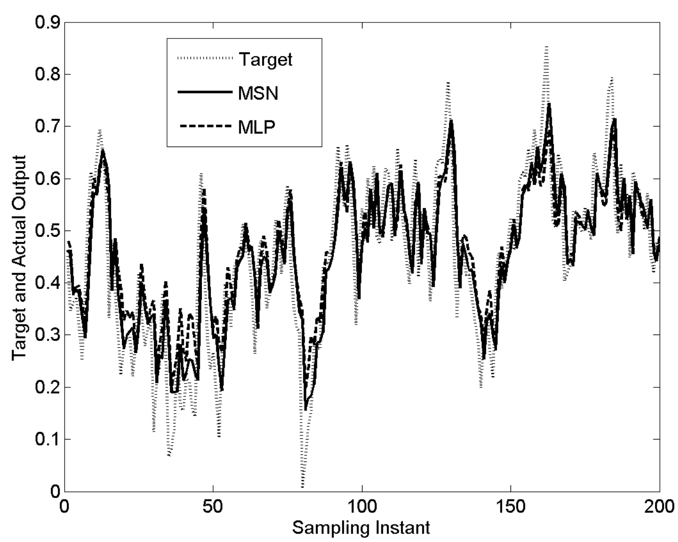


Figure 10. Target and Actual Output with MLP and MSN, for EEG Data

Table 7. Comparison of training and testing performance for EEG data

| S. No. | Parameter | MLP | MSN |
|---|---|---|---|
| 1 | Training Goal, in terms of MSE | 0.013 | 0.010 |
| 2 | Iterations Needed | 750 | 750 |
| 3 | Training Time in seconds | 0.83 | 0.50 |
| 4 | Testing Time in seconds | 0.07 | 0.05 |
| 5 | MSE for Testing Data | 0.0703 | 0.0034 |
| 6 | Correlation Coefficient | 0.9948 | 0.9978 |
| 7 | Number of Parameters | 32 | 16 |

## 5. CONCLUSIONS

A spiking neuron based learning technique is developed. The proposed model incorporates multiplicative aggregation at dendrites and threshold variability in biological neurons. The training and testing results with different benchmark and real-life problems are discussed. It is found that the proposed artificial neural system with a single neuron is capable of performing classification and function approximation tasks as efficiently as a multilayer perceptron with several neurons. Moreover, in some cases its learning is even better than that of a multilayer perceptron. It is also observed that training time and testing time in case of IFN are significantly less as compared with MLP. It is also observed that training and testing times in case of MSN are significantly less as compared with MLP. However, the proposed model cannot be used for multi-output functional mappings. Future research is required for analytical investigation of its capability as a *universal function-approximator*.

## REFERENCES


Bohte, S.M. Poutre, H.L. and Kok, J.N. (2000), SpikeProp: Error-Backpropagation for Networks of Spiking Neurons, Proc. of ESANN, 419-425.

Durbin, R., and Rumelhart, D.E. (1989) “Product units: A computationally powerful and biologically plausible extension to back-propagation networks”, *Neural Computation*, **1**, 133-142.

Feng, J., Buxton, H., and Deng, Y.C. (2002) “Training the integrate-and-fire model with the Informax principle I”, *J. Phys. A*, **35**, 2379-2394.

Feng, J., Sun, Y., Buxton, H., and Wei, G. (2003) “Training integrate-and-fire neurons with the Informax principle II”, *IEEE Transactions on Neural Networks*, **14**, 326-336.

Feng, J. and Li, G. (2001) “Neuronal models with current inputs”, *J. Phys. A*, **24**,1649-1664.

Freeman, W.J. (1988), Why Neural Networks dont Yet Fly: Inquiry into the Neurodynamics of Biological Intelligence, in Proc. of IEEE International Conference on Neural Networks, **2**, 1-7.

Gerstner, W. (1995) “Time structure of the activity in neural network models”, *Phys. Rev. E*, **51**(1), 738-758.

Gerstner, W. and Kistler, W. (2002) *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, Cambridge University Press.

Hebb, D. (1949) *Organization of Behavior*, John Weiley and Sons, New York.

Hodgkin, A.L. and Huxley, A.F. (1952) “A quantitative description of membrane current and its application to conduction and excitation in nerve”, *Journal of Physiology*, **117**, 500–544.

Hopfield, J.J. (1995) “Pattern recognition computation using action potential timing for stimulus representation”, *Nature*, **376**, 33-36.

Koch, C. (1999) *Biophysics of Computation: Information Processing in Single Neurons*, Oxford University Press.

Koch, C. and Poggio, T. (1992) *Multiplying with synapses and neurons”, Single Neuron Computation*, Academic Press: Boston, Massachusetts.

Maass, W. (1997) “Networks of spiking neurons: the third generation of neural network models”, *Neural Networks*, **10**, 1659-1671.

McCulloch, W. and Pitts, W., (1943) “A logical calculus of the ideas immanent in nervous activity”, *Bulletin of Mathematical Biophysics,***5** 115-133.

Mishra, D. Yadav, A. and Kalra, P.K. (2004), Chaotic Behavior in Neural Networks and FitzHugh-Nagumo Neuronal Model, Proceedings of ICONIP-2004, LNCS 3316, India, 868-873.

Rosenblatt, F. (1958) "The perceptron: A probabilistic model for information storage and organization in the brain", Psychol. Review, **65**, 386-394.

Scholles, M.B. Hosticka, J. Kesper, M. Richert, P. and Schwarz, M. (1993), Biologically Inspired Artificial Neurons: Modeling and Applications", Proceedings of International Joint Conference on Neural Networks, IJCNN '93-Nagoya, **3**, 2300-2303.

Sinha, M., Chaturvedi, D.K. and Kalra, P.K. (2002) "Development of flexible neural network", *Journal of IE(I)*, **83**.

Tuckwell, H.C. (1988) *Introduction to Theoretical Neurobiology*, Cambridge University Press.

Widrow, B. and Hoff, M.E. (1960) *Adaptive Switching Circuits,* IREWESCON Connection Recors, IRS, New York.

Widrow, B. and Steams, S. (1985) *Adaptive signal processing*, Prentice-Hall, Englewood Cliffs, NJ.

Yadav, R.N. Singh, V. and Kalra, P.K. (2003), Classification using Single Neuron, Proceedings of IEEE International Conference on Industrial Informatics, Canada, 124-129.

Yadav, A. Mishra, D., Yadav, R.N. Ray, S. and Kalra, P.K. (2005), Learning with Single Integrate-and-Fire Neuron, Proc. of IEEE International Joint Conference on Neural Network, Montreal, 2156 – 2161.

**Received:** August 27th 2005

**Accepted in final format:** January 29th 2007 after one revision.

**About the authors**

*Deepak Mishra* was born in Seoni (Madhya Pradesh, India) on 18th July, 1978. He did his Ph. D. in Electrical Engineering at the Indian Institute of Technology Kanpur, India. His major field of study is Neural Networks, Image Processing, and Computational Neuroscience.Mr. Mishra is a student member of IEEE society. (Home page:http://dkmishra.googlepages.com).

*Abhishek Yadav* was born in Mainpuri (Uttar Pradesh, India) on 21st October, 1976. He did his M. Tech. in Electrical Engineering at the Indian Institute of Technology Kanpur, India in the year 2005. He is working as Assistant Professor in the Department of Electrical Engineering, College of Technology, G. B. Pant University of Agriculture and Technology, Pantnagar, Uttaranchal, India. His major field of study is Computational Neuroscience.

*Sudipta Ray* was born in Rishra (West Bengal, India) on 3rd September, 1980. He did his M. Tech. in Electrical Engineering at the Indian Institute of Technology Kanpur, India. His major field of study is Computational Neuroscience.

*Prem K. Kalra* was born in Agra (Uttar Pradesh, India) on 25th October , 1956. He received his BSc (Engg.) degree from DEI Agra, India in 1978, M.Tech degree from the Indian Institute of Technology, Kanpur, India in 1982 and Ph.D. degree from Manitoba University, Canada in 1987. Presently he is Professor and Head of the Department of Electrical Engineering, IIT Kanpur. He has published over 150 papers in reputed National and International journals and conferences. His research interests are Expert Systems applications, Fuzzy Logic, Neural Networks and Power Systems.