Time-Series Prediction with Single Integrate-and-Fire Neuron

A. Yadav, D. Mishra, R. N. Yadav, S. Ray & P. K. Kalra

Department of Electrical Engineering
Indian Institute of Technology, Kanpur, India
E-mail: {ayadav.iitk@gmail.com, dkmishra@iitk.ac.in}

Abstract

In this paper, a learning algorithm for a single Integrate-and-Fire Neuron (IFN) is proposed and tested for various applications in which a multilayer perceptron neural network is conventionally used. It is found that a single IFN 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 time-series prediction have been illustrated. It is observed that the inclusion of some more biological phenomenon in an artificial neural network can make it more powerful.

Key words: backpropagation, integrate-and-fire neuron, time-series prediction

1 Introduction

Various researchers have proposed many neuron models for artificial neural networks. Although all of these models were primarily inspired from the bi-
ological neuron, there is still a gap between the philosophies used in neuron models for neuroscience studies and neuron models used for artificial neural networks. Some of these models exhibit a close correspondence with their biological counterparts while others do not. Freeman [1] 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, and 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 its ability to represent biological activities with its maximum intricacy, but 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 use in the applications of classification, time-series prediction, function approximation etc. In this work, we used the simplest biological neuron model i.e. integrate and fire model for this purpose.

The first artificial neuron model was proposed by McCulloch and Pitts [2] in 1943. They developed this neuron model based on the fact that the output of the neuron is 1 if the weighted sum of its inputs is greater than a threshold value and 0 otherwise. In 1949, Hebb [3] proposed a learning rule that became initiative for artificial neural networks. He postulated that the brain learns by changing its connectivity patterns. Widrow and Hoff [4] in 1960 presented the most analyzed and most applied learning rule. It was called the least mean square learning rule. Later in 1985, Widrow and Sterns [5] found that this rule converges in the mean square to the solution that corresponds to least
mean square output error if all the input patterns are of the same length. A single neuron of all the above and many other neuron models proposed by several scientists and researchers are capable of linear classification [6]. Yadav, Singh and Kalra [7], in 2003, incorporated various aggregation and activation functions to model the nonlinear input-output relationships. In 2004, Mishra, Yadav and Kalra [8] investigated the chaotic behavior in neural networks that represent biological activities in terms of firing rates. Scholles [15] discussed biologically inspired artificial neurons and Feng and Li [14] introduced neuronal models with current inputs. Training the integrate-and-fire model with the Informax principle was discussed in [12] and [13]. In the present work, a more biologically realistic artificial neural network is proposed and discussed.

2 Biological Neurons

2.1 Architecture of a Biological Neuron

A neuron is the fundamental building block of the biological neural network. A typical neuron has three major regions: the soma, the axon and the 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—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. The schematic diagram of a biological neuron is shown in Figure 1.
2.2 Hodgkin-Huxley and Integrate-and-Fire Neuron Models

Hodgkin and Huxley [19,20] introduced the four-dimensional neuron model which is expressed by the following set of differential equations-

\[
C \frac{dv}{dt} = I_{Na} + I_{K} + I_{L} + I_{EXT} \tag{1}
\]

\[
x \frac{dx}{dt} = -x + x_{\infty} \tag{2}
\]

where,

\[
I_{Na} = G_{Na}m^{3}h(V_{Na} - v) \tag{3}
\]

\[
I_{K} = G_{K}n^{4}(V_{K} - v) \tag{4}
\]

\[
I_{L} = G_{L}(V_{L} - v) \tag{5}
\]
and $x = n, m, h$. $I_{\text{EXT}}$ is the external current injected into the neuron. $I_{Na}$, $I_{K}$ and $I_{L}$ are sodium, potassium and leakage currents, respectively. $G_{Na}$, $G_{K}$ and $G_{L}$ are sodium, potassium and leakage conductances, respectively. $V_{Na}$, $V_{K}$ and $V_{L}$ represent sodium equilibrium, potassium equilibrium and leakage resting potentials, respectively. $n$, $m$ and $h$ are gating variables.

Although integrate and-fire model is a very simple model, it captures some of the major electrical features of a neuron. The basic circuit of this model consists of a capacitor $C$ in parallel with a resistor $R$ driven by a current $I_{\text{EXT}}$. Circuit diagram of an integrate-and-fire neuron model is shown in Figure 2. The driving current can be split into two components, $I_{\text{EXT}} = I_R(t) + I_C(t)$. The first component is the resistive current which passes through the linear resistor $R$ and the second component charges the capacitor $C$. Thus

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

(Fig. 2. Circuit diagram of an integrate-and-fire neuron model)
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 sub threshold and the supra threshold. The Hodgkin-Huxley equations show that in sub threshold 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 (6). In case of the supra threshold 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 3 shows the response of an integrate-and-fire model. The solution of the first-order differential equation describing the dynamics of this model in sub-threshold region can be found analytically. With \( v(0) = V_{RESET} \), the solution is given by

\[
v(t) = \frac{I_{EXT}}{G_L} \left( 1 - e^{-\frac{t}{C/G_L}} \right) + (V_{REST} - V_L) e^{-\frac{t}{C/G_L}} + V_L \tag{7}
\]

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

\[
V_{TH} = \frac{I_{EXT}}{G_L} \left( 1 - e^{-\frac{T_{TH}G_L}{C}} \right) + (V_{REST} - V_L) e^{-\frac{T_{TH}G_L}{C}} + V_L \tag{8}
\]

Therefore, \( T_{TH} \) can be written as

\[
T_{TH} = \frac{C}{G_L} \ln \left( \frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L)} \right) \tag{9}
\]
Interspike interval $T_{ISI}$ is the summation of $T_{TH}$ and $T_{REF}$. Thus

$$T_{ISI} = T_{TH} + T_{REF}$$  \hfill (10)

Therefore,

$$T_{ISI} = \frac{C}{G_L} \ln \left( \frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L)} \right) + T_{REF}$$  \hfill (11)

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

$$f = \frac{1}{\frac{C}{G_L} \ln \left( \frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L)} \right) + T_{REF}}$$  \hfill (12)

![Diagram](image.png)

Fig. 3. Response of an integrate-and-fire neuron model

### 3 The Proposed Model

The Multilayer Perceptron (MLP) is the most popular neuron model that is used in feedforward artificial neural networks. This model represents following
two neuronal activities, one taking place in dendrites and the other in soma. Dendrites work as input devices and are represented in terms of aggregation function which is considered to be the weighted summation in case of MLP. Soma is considered as the main processing element and is represented by a nonlinear transfer function such as log-sigmoid.

In our proposed neuron model, instead of the weighted sum of a conventional neuron (MLP), a new aggregation function is used. This aggregation function is inspired from the relationship between injected current and interspike interval (equation 11) for integrate-and-fire neuron. The expression of interspike interval $T_{ISI}$ is rewritten in Equation 13.

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

(13)

where

$C = $ Membrane capacitance

$G_L = $ Leakage conductance of membrane

$I_{EXT} = $ External (injected) current
For \( n \) number of inputs from different synapses, \( T_{ISI} \) for postsynaptic neuron is associated with nonlinear dendritic interaction. In view of evidences in support of presence of multiplicative-like operations in the nervous system [9,10], multiplication operation is considered to represent this interaction. Thus, for \( n \) number of inputs the input-output relation can be written by modifying equation 13 as 14.

\[
T_{ISI} = \prod_{i=1}^{n} \left( \frac{C_i}{G_{L,i}} \left( \ln (b_i I_{EXT,i} - K_i) - \ln (b_i I_{EXT,i} - L_i) + T_{REF,i} \right) \right)
\]

(14)

where \( b_i \) is the synaptic strength, \( K_i = G_{L,i}(V_{REST,i} - V_{L,i}) \) and \( L_i = G_{L,i}(V_{TH,i} - V_{L,i}) \). Subscript \( i \) is used to denote the respective quantities corresponding to the \( i \)-th synaptic input. Equation 14 is modified to a simplified form, which extracts only the functional form of this equation. It then becomes

\[
x_{\text{net}} = \prod_{i=1}^{n} (a_i \ln(b_i x_i) + d_i)
\]

(15)

where, \( n \) is the number of inputs.

Equation 15, corresponds to the first part of integrate-and-fire model which is represented by an RC circuit in Figure 2. In Equation 15, net input to the second part of neuron model i.e., \( x_{\text{net}} \) is considered to represent the interspike interval. The \( i \)-th input i.e., \( x_i \) in equation 15 is considered to be analogous to the injected current \( I_{EXT,i} \) and it is given in 16. Parameter \( a_i \) represents
the time constant $C_i/G_{L,i}$, $b_i$ is the synaptic strength and $d_i$ is assumed to be associated with the input $x_i$ to represent spatial and temporal summation when inputs from several synapses are present.

$$x = \frac{I_{EXT} - G_L(V_{REST} - V_L)}{I_{EXT} - G_L(V_{TH} - V_L)}$$

In biological neural systems, input-output relationship depends on the timings of various spikes which are approximated in terms of exponential functions [18]. Aggregation of exponential waveforms with different time-delays has been approximated by considering different weights associated with the input to the first block representing the aggregation function.

The second part of the integrate-and-fire neuron model is represented in terms of a threshold type of nonlinear block (figure 2). A sigmoid function given in Equation 17 is considered to represent the activity in this block. This function is used as the activation function of the proposed neuron model.

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

### 3.1 Biological Significance of the Proposed Model

This model is inspired from the hypothesis that the actual shape of the action potential does not contain any neuronal information; it is the timing of spikes that matters. As the firing frequency is directly related to the injected current, we considered the f-I characteristic of integrate-and-fire neuron as the backbone of our model.

All artificial neuron models have two functions associated with them: aggre-
ation and activation. In case of integrate-and-fire model also, there are two parts in its circuit representation: an RC-circuit and a threshold-type nonlinearity (figure 2). While aggregation is inspired from the f-I relation derived from the response of the RC circuit (equation 12), nonlinearity is introduced in terms of sigmoid activation function (equation 17). This activation function is continuous and differentiable, therefore it can easily be incorporated in learning. As its output $f(x)$ approaches zero when input $x$ approaches a large value, and is always greater than 0.9933, i.e., $x > 5.0$, it can be considered to represent an approximation of threshold-type nonlinearity to some extent. A substantial body of evidence supports the presence of multiplicative-like operations in the nervous system [9]. Physiological and behavioral data strongly suggest that the optomotor response of insects to moving stimuli is mediated by a correlation-like operation [10]. Another instance of a multiplication-like operation in the nervous system is the modulation of the receptive field location of neurons in the posterior parietal cortex by the eye and head positions of the monkey [10]. Multiplication operation is used for aggregation of inputs to the artificial neuron in many research papers including [7]. In our work, we incorporated this multiplication operation while aggregating inputs to the activation function.

3.2 Development of the Training Algorithm

Training algorithms for an artificial neural network are nothing but optimization techniques. As the existing optimization techniques are sufficiently powerful, we used the most popular training algorithm, i.e., backpropagation, for the comparison of our neuron model with the existing one. A simple steepest
A descent method is applied to minimize the following error function:

\[ e = \frac{1}{2}(y - t)^2 \]  

(18)

where, \( t \) is the target and \( y \) is the actual output of the neuron. \( e \) is the function of parameters \( a_i, b_i \) & \( d_i; \ i = 1, 2, \ldots n \). Therefore, the parameter update rule (weight update rule) can be expressed in terms of the following equations:

\[ a_i^{\text{new}} = a_i^{\text{old}} - \eta \frac{\partial e}{\partial a_i} \]  

(19)

\[ b_i^{\text{new}} = b_i^{\text{old}} - \eta \frac{\partial e}{\partial b_i} \]  

(20)

\[ d_i^{\text{new}} = d_i^{\text{old}} - \eta \frac{\partial e}{\partial d_i} \]  

(21)

For \( i = 1, 2, 3, \ldots n; \) where, \( \eta \) is the learning rate. Partial derivatives of \( e \) with respect to parameters \( a_i, b_i \) & \( d_i; \ i = 1, 2, \ldots n \) can be given by the following equations:

\[ \frac{\partial e}{\partial a_i} = (t - y)y(1 - y)\text{net} \left( \frac{\log(b_i x_i)}{a_i \log(b_i x_i) + d_i} \right) \]  

(22)

\[ \frac{\partial e}{\partial b_i} = (t - y)y(1 - y)\text{net} \left( \frac{1}{a_i \log(b_i x_i) + d_i} \right) \left( \frac{a_i}{b_i} \right) \]  

(23)

\[ \frac{\partial e}{\partial b_i} = (t - y)y(1 - y)\text{net} \left( \frac{1}{a_i \log(b_i x_i) + d_i} \right) \]  

(24)
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 integrate-and-fire neuron (IFN) with that of multilayer perceptron (MLP). For this purpose, we considered an MLP with 3 hidden units in a single hidden layer. Figure 5 shows the mean-square-error (MSE) vs. number of epochs curves for training with MLP and IFN while dealing with the XOR-problem. It is clear from this figure that the proposed model takes only 31 iterations while MLP takes 523 iterations for training to achieve an MSE of the order of 0.0001. Table 1 exhibits the comparison between MLP and IFN in terms of the deviation of actual outputs from corresponding targets. It can be seen here that the performance of IFN is almost same as compared with MLP. From Table 1, it is observed that the training time required by IFN is much less than MLP. It means that a single IFN is capable to learn XOR relationship almost 4 times faster than an MLP with 3 hidden units. Table 2 shows the comparison of training and testing performance with MLP and IFN while solving the XOR-problem.
Table 1

Outputs of IFN and MLP for XOR-problem

<table>
<thead>
<tr>
<th>Input</th>
<th>Target</th>
<th>Actual Output with MLP</th>
<th>Actual Output with IFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>0</td>
<td>0.0001</td>
<td>0.0011</td>
</tr>
<tr>
<td>0 1</td>
<td>1</td>
<td>0.9890</td>
<td>0.9852</td>
</tr>
<tr>
<td>1 0</td>
<td>1</td>
<td>0.9922</td>
<td>0.9841</td>
</tr>
<tr>
<td>1 1</td>
<td>0</td>
<td>0.0204</td>
<td>0.0109</td>
</tr>
</tbody>
</table>

Fig. 5. Mean Square Error vs. Epochs for training for XOR-problem

4.1.2 3-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 1’s then the parity is 1, otherwise it is 0. This is considered as 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 IFN and MLP in case of 3-bit Parity Problem the optimum configuration of MLP is selected for comparison and
Table 2

Comparison of testing and training performance for XOR-problem

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Parameter</th>
<th>MLP</th>
<th>IFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Training Goal, in terms of MSE</td>
<td>0.00015</td>
<td>0.00015</td>
</tr>
<tr>
<td>2</td>
<td>Iterations Needed</td>
<td>523</td>
<td>31</td>
</tr>
<tr>
<td>3</td>
<td>Training Time in seconds</td>
<td>0.54</td>
<td>0.13</td>
</tr>
<tr>
<td>4</td>
<td>Testing Time in seconds</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>5</td>
<td>MSE for Testing Data</td>
<td>0.0001495</td>
<td>0.0001479</td>
</tr>
<tr>
<td>6</td>
<td>Correlation Coefficient</td>
<td>0.999890</td>
<td>0.999975</td>
</tr>
<tr>
<td>7</td>
<td>Percentage Misclassification</td>
<td>0 %</td>
<td>0 %</td>
</tr>
<tr>
<td>8</td>
<td>Number of Parameters</td>
<td>11</td>
<td>6</td>
</tr>
</tbody>
</table>

therefore, we considered MLP with 5 hidden units in a single hidden layer.

Figure 6 shows the comparison of MSE vs. number of epochs curves with conventional multilayer perceptron and the proposed single IFN model while training the artificial neural systems for 3-bit Parity Problem. It is clear from this figure that the proposed model takes only 15 iterations as compared to 837 iterations taken by MLP for training to achieve an MSE of the order of 0.001. Table 3 exhibits the comparison between MLP and IFN in terms of the deviation of actual outputs from corresponding targets. It can be observed that the performance of IFN is almost same as compared with MLP but IFN is capable to learn this relationship almost 6 times faster than that in case of an MLP with 5 hidden units in a single hidden layer. Table 4 shows the comparison of training and testing performance with MLP and IFN while solving the 3-bit Parity problem.
Table 3

Outputs of IFN and MLP for 3-bit parity problem

<table>
<thead>
<tr>
<th>Input</th>
<th>Target</th>
<th>Actual Output with MLP</th>
<th>Actual Output with IFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0 0</td>
<td>0</td>
<td>0.1002</td>
<td>0.0603</td>
</tr>
<tr>
<td>0 0 1</td>
<td>1</td>
<td>0.9890</td>
<td>0.9952</td>
</tr>
<tr>
<td>0 1 0</td>
<td>1</td>
<td>0.9922</td>
<td>0.9949</td>
</tr>
<tr>
<td>0 1 1</td>
<td>0</td>
<td>0.0304</td>
<td>0.0549</td>
</tr>
<tr>
<td>1 0 0</td>
<td>1</td>
<td>1.0000</td>
<td>0.9707</td>
</tr>
<tr>
<td>1 0 1</td>
<td>0</td>
<td>0.0822</td>
<td>0.0152</td>
</tr>
<tr>
<td>1 1 0</td>
<td>0</td>
<td>0.0091</td>
<td>0.0008</td>
</tr>
<tr>
<td>1 1 1</td>
<td>1</td>
<td>0.9374</td>
<td>0.8846</td>
</tr>
</tbody>
</table>

Table 4

Comparison of testing and training performance for 3-bit Parity problem

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Parameter</th>
<th>MLP</th>
<th>IFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Training Goal, in terms of MSE</td>
<td>0.003</td>
<td>0.003</td>
</tr>
<tr>
<td>2</td>
<td>Iterations Needed</td>
<td>837</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>Training Time in seconds</td>
<td>0.63</td>
<td>0.11</td>
</tr>
<tr>
<td>4</td>
<td>Testing Time in seconds</td>
<td>0.02</td>
<td>0.02</td>
</tr>
<tr>
<td>5</td>
<td>MSE for Testing Data</td>
<td>0.0055</td>
<td>0.0053</td>
</tr>
<tr>
<td>6</td>
<td>Correlation Coefficient</td>
<td>0.9977</td>
<td>0.9969</td>
</tr>
<tr>
<td>7</td>
<td>Percentage Misclassification</td>
<td>0 %</td>
<td>0 %</td>
</tr>
<tr>
<td>8</td>
<td>Number of Parameters</td>
<td>22</td>
<td>9</td>
</tr>
</tbody>
</table>
4.2 Time-Series Prediction 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. The problem is intentionally selected as it is observed that it is not predictable with linear models. Four measurements $y(t-1)$, $y(t-2)$, $y(t-4)$ and $y(t-8)$ were used to predict $y(t)$. These four measurements are selected by comparing the performance with a few alternatives on trial basis. For comparison of the performance with IFN and MLP with Internet Traffic Data, we considered MLP with 6 hidden neurons in a single hidden layer. It is to be noted here that the optimum configuration of MLP is selected for comparison. Figure 7 shows the comparison of MSE vs. number of epochs curves while training MLP and IFN artificial neural systems for Internet Traffic Data. This figure shows that the proposed model exhibits faster training on this data. Figure 8 shows the comparison between MLP and IFN in terms of the deviation of actual outputs from corresponding targets.
Data till 200 sampling instants was used for training and rest of the data was used for testing. It can be observed that the performance of IFN for training data is almost same as compared with MLP while its performance is better for testing data i.e., after 200 sampling instants. **In terms of training time, it is also observed that IFN can be trained almost 3 times faster than MLP for this data.** Table 5 shows the comparison of training and testing performance with MLP and IFN for the *Internet Traffic Data*.

![Mean Square Error vs. Epochs for training for Internet Traffic Data](image1)

**Fig. 7. Mean Square Error vs. Epochs for training for Internet Traffic Data**

![Target and Actual Output with MLP and IFN, for Internet-Traffic Data](image2)

**Fig. 8. Target and Actual Output with MLP and IFN, for Internet-Traffic Data**
Table 5

Comparison of testing and training performance for Internet-Traffic Data

<table>
<thead>
<tr>
<th>S. No.</th>
<th>Parameter</th>
<th>MLP</th>
<th>IFN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Training Goal, in terms of MSE</td>
<td>0.005</td>
<td>0.005</td>
</tr>
<tr>
<td>2</td>
<td>Iterations Needed</td>
<td>200</td>
<td>8</td>
</tr>
<tr>
<td>3</td>
<td>Training Time in seconds</td>
<td>0.73</td>
<td>0.24</td>
</tr>
<tr>
<td>4</td>
<td>Testing Time in seconds</td>
<td>0.08</td>
<td>0.06</td>
</tr>
<tr>
<td>5</td>
<td>MSE for Testing Data</td>
<td>0.0214</td>
<td>0.0013</td>
</tr>
<tr>
<td>6</td>
<td>Correlation Coefficient</td>
<td>0.9973</td>
<td>0.9996</td>
</tr>
<tr>
<td>7</td>
<td>Number of Parameters</td>
<td>32</td>
<td>12</td>
</tr>
</tbody>
</table>

Fig. 9. Mean Square Error vs. Epochs for training for EEG Data

4.2.2 Electroencephalogram Data

Electroencephalogram (EEG) data used in this work was taken from [21]. Presence of randomness and chaos [8] in this data makes it interesting for neural network related research. This problem is intentionally selected as it is observed that it is not predictable with linear models. In this problem also,
four measurements $y(t - 1)$, $y(t - 2)$, $y(t - 4)$ and $y(t - 8)$ were used to predict $y(t)$. These four measurements are selected by comparing the performance with a few alternatives on trial basis. The optimum configuration of MLP is selected for comparison. We considered MLP with 6 hidden neurons in a single hidden layer for the comparison purposes. Figure 9 shows the comparison of MSE vs. number of epochs curves with conventional multilayer perceptron and
the proposed single IFN model while training the artificial neural systems for EEG data. It is clear from this figure that the proposed model exhibits faster training on this data. Figure 10 shows the comparison between MLP and IFN in terms of the deviation of actual outputs from corresponding targets. Data till 100 sampling instants was used for training. It can be seen here that the performance of IFN for training as well as testing data is much better than that of MLP. It means that a single IFN is capable to learn this relationship faster than that in case of an MLP with 7 neurons in a hidden layer and its performance on seen as well as unseen data is significantly better. Table 6 shows the comparison of training and testing performance with MLP and IFN while applying for the EEG Data.

5 Conclusions

The training and testing results with different benchmark and real-life problems show that the proposed artificial neural system with a single neuron inspired from the integrate-and-fire neuron model is capable of performing classification and function approximation tasks as efficiently as a multilayer perceptron with many neurons and in some cases its learning is even better than that of a multilayer perceptron. It is also observed that training and testing times in case of IFN are significantly less as compared with MLP. Future scope of this work includes incorporation of these neurons in a network and analytical investigation of its learning capabilities (e.g., as universal function approximator).
References


[21] www.cs.colostate.edu/eeg/