Comparative Study of Single Biological Neuron with an Artificial Neuron

Sashmita Panda

G.S.Sanyal School of Telecommunications, IIT Kharagpur, India
E-mail: ysashmita786@gmail.com

Abstract. A number of artificial neural models have been presented in the literature in an effort to suggest a more accurate representation of a single biological neuron. There are numerous publications on synthetic neurons that attempted to replicate a single biological neuron, however, such models were unable to generate the spiking patterns of a real biological neuron. Therefore, there is still scope to design and research improved spiking neural models that more accurately reflect the functions of a biological neuron. This motivation drives extensive modification of an artificial neuron model to produce the spike patterns of a real biological neuron. The modified single artificial neuron model that has been proposed exhibits the functions of a biological neuron. It’s still crucial to model spiking bio-neuron behavior. Modeling a spiking bio-neuron is still an important exercise in view of possible applications of the underlying features in the areas of neuromorphic engineering, cognitive radio, and spiking neural networks.

Keywords: Artificial Neuron, Biological Neuron, Artificial neural network (ANN), Back Propagation (BP) Algorithm, Spike response model.

INTRODUCTION

A biological neuron is the basic dynamic element of the human central nervous system. When a neuron is triggered above a certain threshold value, it produces a brief electrical pulse known as a “spike” [1]. The synapse, a fluidic area, is where a spike is transferred from one neuron to another [2]. In response to chemical and other inputs, neurons are highly specialized for producing and sending electrical signals [3]. A neuron model that accurately reflects the biological characteristics of the Hodgkin-Huxley model and the computational effectiveness of the integrate-and-fire model is provided in [4]. This model reproduces the spiking and bursting behavior of recognized types of cortical neurons. Neurotransmitters and electrochemical impulses help neuro-spike communication. At an axonal terminal of a neuron, synaptic vesicles with neurotransmitters are found. The synaptic cleft is a tiny fluid-filled space between an axon terminal of a pre-synaptic neuron and a spine of a different post-synaptic neuron. Voltage-gated $\text{Ca}^{2+}$ channels become active in response to action potential or spike reaching a presynaptic neuron’s axon terminal, allowing $\text{Ca}^{2+}$ ions to enter. With the aid of $\text{Ca}^{2+}$ ions, the vesicles are fused to the neuronal membrane, and finally, neurotransmitters are released into the synaptic cleft.

Numerous studies are currently being conducted on how our brain’s neurons represent stimuli, and it has been suggested that the timing of the action potentials or spikes that these neurons release contains information [5]. Reaching bio-inspired nanoscale paradigms requires a basic understanding of neuro-spike transmission, which is the basic mode of communication between neurons. In the paper [6], the author developed a plausible model to explain neuro-spike communication. Hodgkin and Huxley explained how variations in the conductance of $\text{Na}^+$ and $\text{K}^+$ in the axon membrane can be used to create ionic currents in the giant squid axon [7]. They built a mathematical model based on the voltage and time-dependent characteristics of the $\text{Na}^+$ and $\text{K}^+$ conductance by carrying out several voltage clamp tests [8]. Through this study, a system of differential aligns was developed, eventually referred to as the Hodgkin-Huxley model [9], which defined the ionic foundation of the action potential.

Some cortical neurons spiking and bursting behavior is described by the model [4]. It combines the computational efficiency of the integrate-and-fire neuron with the dynamics of the Hodgkin-Huxley model. This model allows for...
the real-time simulation of tens of thousands of spiking cortical neurons. This model is developed in paper [10] using first-order log domain low pass filters and two translinear multipliers. The spiking patterns of this chip-created neuron model could be observed by altering the input current as well as the biased voltages and currents. The Mihalas-Niebur neuron model is also known as the generalized integrate-and-fire neuron model [11]. Many of the spiking and bursting characteristics displayed by this model are observed in actual biological neurons [10, 12]. It uses straightforward first-order differential aligns to explain each of the state variables, in contrast to other simplified Hodgkin-Huxley neuron models [13]. This neuron model has a number of benefits, including the ability to be bio-physically interpreted, which enables one to understand what might occur in biological neurons.

Additionally, it offers a methodical technique to incorporate numerous additional mechanisms and state variables. The leaky-integrate and fire (LIF) model’s counterpart, the spike response model [14], likewise explains how neurons produce action potentials. While integrate-and-fire versions are based on differential aligns for the membrane potential, the spike response model relies primarily on filters. This paper discussed a neuron model that accurately simulated the membrane voltage dynamics of a biological neuron cell by including a changeable leaky resistor and bias current [15]. The author in [16] examines a straightforward model that can faithfully depict the spiking behavior of a neuron. While capturing a sizable percentage of the complexity of biophysical models, the spike train permits the development of models that are significantly less complex than those models. A mathematical description of the evolution of membrane potentials and an adaptation current is provided by a two-dimensional neuron model [14, 17]. It is an evolution of the exponential LIF neuron that imitates the upswing with an exponential function and the downswing with a reset condition in an action potential. The exponential term causes the voltage to increase quickly when the membrane potential approaches the threshold voltage [18].

Some of the parameters used in this model also affect subthreshold adaptation and spike-triggered adaptation. As a result of biological models, numerous artificial models have been created. The first neural network model [19], which uses directed weighted routes to connect the neurons, was described by the author in the study [20]. A generalized model for giving the threshold in the axon with nonlinear dynamics that depend on time is the McCulloch-Pitts neuron [21]. The linear sum of the weighted inputs from the other neurons in the network determines the binary unit’s value in this model [22]. To represent the network of artificial neurons, researchers have built numerous network models. Current applications of artificial neural networks (ANN) span a range of social, industrial, financial, and scientific contexts. Among the frequently used techniques in these fields are a functional approximation, filtering, direct modeling or system identification, inverse modeling or channel equalization, control, classification, forecasting, pattern recognition, and optimization. Rumelhart invented the conventional backpropagation (BP) technique of multilayer artificial neural network (MANN) as a supervised learning strategy [23, 24] which is a gradient descent local optimization technique.

### DESCRIPTION OF SINGLE BIOLOGICAL NEURON MODEL

The block diagram of a single biological neuron model presented in Fig. 1 comprises some inputs at the dendritic end. Each of \( j \)th \((1 \leq j \leq J)\) input of \( i \)th sequence, \( x_{ij} \) is multiplied with its associated gamma-distributed time-varying synaptic weights, \( w_{ij}(t) \) to produce the required output \( u_i(t) \). In this case, each \( x_{ij} \) represents either a ‘0’ or ‘1,’ and each input sequence consists of a \( J \) number of bits. The strength of the connection between two neurons varies in practice and is represented as synaptic weight. Either the height of the postsynaptic potential or the slope of the postsynaptic current denotes the amplitude response and is determined by weights. In most of the single neuron models, long-term synaptic plasticity is employed, which assumes constant synaptic weights [25].

In Fig. 1, \( u_i(t) \), which denotes the membrane potential. If it crosses some threshold value, then action potentials in terms of spikes are generated. Here \( u_i(t = 0) \) is the initial membrane potential at the beginning of the \( i \)th observation interval.

The membrane potential, \( u_i(t) \) for \( i \)th input sequence of biological neuron model is computed as

\[
u_i(t) = \sum_{j=1}^{J} x_{ij} w_{ij}(t) + u_i(t = 0) \tag{1}\]

where \( 1 \leq i \leq I, I \) is total number of observation intervals, \( 1 \leq j \leq J, J \) is total number of bits in a sequence, \( x_{ij} \)

![Figure 1. Block diagram of a single biological neuron model.](image-url)
\[ w_{ij}(t) = \frac{t_f}{t_p} h_p f \left( 1 - \frac{t}{t_p} \right) \] (2)

\( f_j \) is the gamma distributed random variable with mean and variance chosen as 0.5 and 0.3 respectively. The symbol \( t_p \) represents the time when \( w_{ij}(t) \) has attained maximum amplitude \( h_p \). In the proposed model, the AMPA receptor is considered which has a typical value of \( t_p = 1\text{msec} \) and \( h_p = 1\text{mV} \) respectively. The output \( y \) of the model can be expressed as;

\[ y = \phi(u(t)) \] (3)

\( \phi(u(t)) = 1 \), when \( u(t) \geq TH \) otherwise \( \phi(u(t)) = 0 \), when \( u(t) < TH \), where ‘TH’ represents a certain output threshold voltage.

**DEVELOPMENT OF A SINGLE ARTIFICIAL NEURON MODEL EQUIVALENT TO BIOLOGICAL NEURON**

A mathematical model of a single artificial neuron which is equivalent to a single biological neuron or a spiking neuron is shown in Fig. 2. The model consists of the inputs \( X_1 \ldots X_j \ldots X_{NB} \). The input sequence is taken in terms of ‘0’s and ‘1’s. Each input is connected with their synaptic weights \( W_1 \ldots W_j \ldots W_{NB} \). Now, the inputs multiplied with synaptic weights are given to summer or an adder which sums all the inputs as a linear combiner. Then, the output of an adder is applied to an activation function which sums all the inputs as a linear combiner. Then, the output ‘y’ of the model can be expressed as;

\[ y = \phi(u(t)) \] (3)

**Steps Followed by the Single Artificial Neuron Model**

In this subsection, a sequence of steps is followed to explain the working principle of a single artificial neuron model which is equivalent to a biological neuron.

**Step 1:** Take the input sequence in terms of ‘0’s and ‘1’s. Here we provide the input sequence as \( X_1 \ldots X_j \ldots X_{NB} \) where \( NB = 20 \).

**Step 2:** Load the outputs obtained from the biological neuron model of vector size (1000x1) and stored them in an excel sheet.

**Step 3:** Assign the random value of weights in the artificial neuron model, then the input sequence will be multiplied with the random weights, and then it will get summed by a summer using the following alignment.

\[ u = \sum_{j=1}^{NB} x_j w_j \] (4)

Where ‘u’ is the output of the summing junction and \( 1 \leq u \leq NB \).

**Step 4:** The summing junction output is applied to a suitable threshold function where a sigmoid function is used as a threshold.

\[ y_1 = \phi(u) = \frac{A_H}{1 + e^{-B(u-TH)}} \] (5)

Where \( y_1 \) is the output of the single artificial neuron model. \( A_H \) is the maximum amplitude of \( y_1 \), B is a scalar quantity and TH is the threshold value after which the output \( y_1 \) decreases.

**Step 5:** Now we can give an input of vector size 1000x20 to the artificial neural model, then in the similar ways explained in step 4 using Eq.(5) we will find the output of the artificial neural model.

**Step 6:** To train the artificial neuron model, we will give the output of the single spiking neuron having vector size 1000x1 which is explained in step 2.

**Training of Artificial Neuron Model**

An artificial neuron model can be trained by the following steps;

**Step 1:** After providing the output of a biological neuron, the model will compute the error function as;

\[ e = f(y_2, y_1) = \frac{1}{2} (y_2 - y_1)^2 \] (6)

Where \( y_2 \) represents the output of the target or desired output, here \( y_2 \) denotes the output of the single biological neuron, and \( y_1 \) is the actual output of the single artificial neuron model.

**Step 2:** Find the error between the output of a biological and artificial neuron model. Now by using a learning rule, this error will be feedback to the artificial neuron model in order to update the weights in accordance to get the actual
output which will be nearly equal to the desired biological neural model output.

**Step 3:** The weights can be updated by using the following learning rule;

$$\Delta W = -\eta \frac{\delta e}{\delta w}$$  \hspace{1cm} (7)

Where $\eta$ is the learning rate parameter and $e$ is the error function of the model and its value lies between 0 to 1.

**Development of Updated Weights for the Single artificial Neuron Model**

In this subsection, the learning rule of updated weights of the proposed AN model is derived. The output of the summing junction is expressed in Eq. (4);

$$u = \sum_{j=1}^{NB} x_j w_j$$

Now, the output of the single artificial neuron model is represented in Eq. (5);

$$y_1 = \phi(u) = \frac{A_H}{1 + e^{-B(u-TH)}}$$

Error function of the model is given in Eq. (6);

$$e = f(y_2, y_1) = \frac{1}{2} (y_2 - y_1)^2$$

Now updating the weights using the learning rule defined in Eq. (7)

$$\Delta W = -\eta \frac{\delta e}{\delta w}$$

By using the chain rule Eq. (7) can be written as follows;

$$\Delta W = -\eta \frac{\delta e}{\delta w} \frac{\delta y_1}{\delta u} \frac{\delta u}{\delta w_j}$$

We know;

$$\frac{\delta e}{\delta y_1} = \frac{1}{2} (y_2 - y_1)^2$$  \hspace{1cm} (8)

$$\frac{\delta y_1}{\delta u} = \frac{\delta}{\delta u} \left( \frac{A_H}{1 + e^{-B(u-TH)}} \right) = \frac{A_H B(u - TH) e^{-B(u-TH)}}{(1 + e^{-B(u-TH)})^2} = A_H B(u - TH) \frac{1 + e^{B(u-TH)}}{(1 + e^{-B(u-TH)})^2}$$  \hspace{1cm} (9)

$$\frac{\delta u}{\delta w_j} = x_j$$

Hence,

$$\Delta w_j = -\eta \frac{\delta e}{\delta w_j} \frac{\delta y_1}{\delta u} \frac{\delta u}{\delta w_j} = \eta (\frac{\delta e}{\delta y_1}) (\frac{\delta y_1}{\delta u}) (\frac{\delta u}{\delta w_j})$$

$$= \eta (y_1 - y_2) A_H B(u - TH) y_1 (1 - y_1) x_j$$  \hspace{1cm} (10)

Where $y = \phi(u) = \frac{A_H}{1 + e^{-B(u-TH)}}$

From the model,

$$\frac{\delta u}{\delta w_j} = x_j$$

$$\Delta w_j = \eta x_j \frac{A_H}{1 + e^{-B(u-TH)}} (y_1 - y_2)$$

If $e = y_1 - y_2$ and $\delta = ey_j (1 - y_1)$

Finally, the change of weights in the $j$th branch is derived as;

$$\Delta w_j = \eta \delta A_H B(u - TH) x_j$$  \hspace{1cm} (11)

Now, using the learning rule,

$$W_{New} = W_{Old} + \Delta W$$

Where $W_{New}$ is a new weight, $W_{Old}$ is the older weight, and $\Delta W$ is the updated weight for the model. Substituting the value of $\Delta w_j$ in the $j$th branch the updated weight can be given as;

$$W_j = w_j + \Delta w_j$$

$$W_j = w_j + \delta A_H B(u - TH) x_j$$  \hspace{1cm} (12)

Where $W_j$ is a new weight and $w_j$ is the old weight for the artificial neuron model.

**Step 4:** Now weight of the model which is determined in step 3 is applied for finding new weights.

**Step 5:** The training procedure will continue through updating the weights till the error function $e$ will be minimum i.e. the output of both single artificial neuron and biological neuron model will be approximately equal to $e$ ($e \leq e$ assume $e = 0.0001$).
RESULTS AND DISCUSSIONS

In this section, the results of output between a single biological neuron (BN) and an artificial neuron (AN) model are analyzed and contributions of the study have been highlighted. For simulating the proposed models a total of 50 input sequences or patterns have been applied to both BN and AN models. By applying a threshold voltage of 15mV to the BN model, specific spike patterns at the output have been observed. The typical numerical values used for the simulation study of the proposed model are listed in Table 1.

For experimental-based simulation study, input random binary sequence each of length J (J = 20) is considered for a time interval of 5 msec. Each input bit is then multiplied with the time-varying synaptic weights obtained at an interval of $T_s$ msec. After each successive $T_s$ msec time interval, all the twenty partial products are added at the summer node to produce an output. After a time interval of 5 msec, the cumulative sum of outputs due to all ten weights are added to produce a membrane potential. When the magnitude of this potential crosses some predefined threshold value, then the neuron gets fired, and a spike is produced in the final output.

It is observed from Fig. 3 that after applying 100 input patterns to a BN, it generates 95 spikes whereas in the AN model it gives 92 spikes shown in Fig. 4. The mean square error reduces to 0.063 for an artificial neuron model presented in Fig. 5 which represents an artificial neuron that can mimic a biological neuron.

Similarly when the number of input patterns increases to 200, then the spikes generated by the BN model is 38 shown in Fig. 6 whereas the AN model produces 40 spikes represented in Fig. 7. From Fig. 8, it is seen that the error in the AN model reduces to 0.066.

Now after increasing the number of input patterns to 500, both the BN model and AN model give 18 spikes which are shown in Fig. 9 and 10 respectively.

It is observed that the mean square error decreases to 0.0047 for the AN model shown in Fig. 11 and hence an

<table>
<thead>
<tr>
<th>Symbols</th>
<th>Description</th>
<th>Typical Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t_p$</td>
<td>The time instant when $w_{ij}(t)$ has maximum amplitude</td>
<td>1msec</td>
</tr>
<tr>
<td>$h_p$</td>
<td>The maximum amplitude of $w_{ij}(t)$ which occurs at $t = t_p$</td>
<td>1mv</td>
</tr>
<tr>
<td>$\mu$</td>
<td>Mean of Gamma distributed random variable</td>
<td>0.5</td>
</tr>
<tr>
<td>$\sigma$</td>
<td>Variance of Gamma distributed random variable</td>
<td>0.3</td>
</tr>
<tr>
<td>$T_s$</td>
<td>Sampling Time</td>
<td>0.5msec</td>
</tr>
</tbody>
</table>

Figure 3. Plot of output for a single biological neuron model for 100 input patterns.

Figure 4. Plot of output for a single artificial neuron model for 100 input patterns.

Figure 5. Plot of error for artificial neuron model for 100 input patterns.
Figure 6. Plot of output for a single biological neuron model for 200 input patterns.

Figure 7. Plot of output for a single artificial neuron model for 200 input patterns.

Figure 8. Plot of error for an artificial neuron model for 200 input patterns.

Figure 9. Plot of output for a single biological neuron model for 500 input patterns.

Figure 10. Plot of output for a single artificial neuron model for 500 input patterns.

Figure 11. Plot of error for an artificial neuron model for 500 input patterns.
artificial neuron functions the same as an artificial neuron. It can be illustrated that with an increase in the number of input patterns to both the BN and AN model the error decreases and the proposed AN model shows the realistic characteristics of a biological neuron. Table 2 illustrates that when the number of input patterns increases, the artificial neuron exhibits a spiking behavior that is quite comparable to that of a true biological neuron.

**CONCLUSION**

An improved biological neuron model is proposed and its output spike patterns are presented in this paper. By applying the same input patterns the response of the biological neuron model is compared with the proposed AN model. The comparison reveals that the mean square error during the training phase of an artificial neural model decreases as the number of input patterns or sequences increases, implying that an artificial neuron model acts more like a biological neuron. The simulation findings also show that the output of the artificial neuron model is almost similar to that of a biological neuron.

**REFERENCES**


---

<table>
<thead>
<tr>
<th>Number of Input Patterns</th>
<th>Number of Spikes in BN Model</th>
<th>Number of Spikes in AN Model</th>
<th>MSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>95</td>
<td>92</td>
<td>0.063</td>
</tr>
<tr>
<td>200</td>
<td>38</td>
<td>40</td>
<td>0.066</td>
</tr>
<tr>
<td>500</td>
<td>18</td>
<td>18</td>
<td>0.0047</td>
</tr>
</tbody>
</table>