Spiking Neural Network on Curve Fitting

Jie-qiong YU, Wen-juan ZHANG and Lei ZHANG*

School of Physics, Northeast Normal University; and National Demonstration Center for Experimental Physics Education (Northeast Normal University), Changchun 130024, P. R. China
*Corresponding author

Keywords: Spiking Neural Network, Curve fitting, MATLAB.

Abstract. Currently, curve fitting has been widely used in data processing. Spiking Neural Network is used to provide fast and accurate curve fitting with discrete data in this paper. First, the principle of Spiking Neural Network is introduced. Second, the Spiking Neural Network for curve fitting based on the MATLAB simulation platform is established. Finally, curve fitting of a linear function and an exponential function are made. The average error values are 0 and 0.3, the results show that Spiking Neural Network can be used in curve fitting effectively.

Introduction

Curve fitting is a method of processing data that uses a continuous curve approximation or matches a set of discrete points on the plane to represent a function between coordinates. Applications of curve fitting have important theoretical and practical significance for revealing the inherent law of the data. However, the traditional methods of fitting curves have some limitations, such as polynomial fitting with low accuracy and cubic spline interpolation with high precision but long computation time. In recent years, artificial neural networks are widely used in information science, brain discipline, neuron psychology and other disciplines. Artificial neural networks[1,2](ANN) are a model that processes distributed parallel information using characteristics of biological nervous system behavior. As a third-generation artificial neural network, Spiking neural network (SNN) represents the latest research results in the field of biological neuron science and ANN. SNN with better biological properties and powerful computing capabilities can simulate a variety of neuronal signals and arbitrary continuous functions[3].

SNN

SNN is a new network proposed by Mass[4]. SNN uses the accurate time of pulse to code information, compared with the traditional way that uses firing frequency of pulses to code information. Therefore, SNN is closer to the processing mechanisms of biological information.

The Model of the Spiking Neuron

The spiking neuron is a mathematical model that is closer to the biological neurons. The traditional Sigmoid neuron model changes a real number input into a real number output signal by passing a transfer function. The essence of the model is, when the spiking neuron receives an external stimulus, its membrane potential will increase. When the membrane potential reaches the threshold, the neuron will generate a spike and send an output signal[5]. There are several common neuron models such as the integrate and fire (I&F) model, (leaky integrate and fire (LIF) model, Hodgkin-Huxley (H-H) model, spike response model (SRM) model. The I & F neuron model is used in this paper, because it can obtain better dynamic characteristics of the nervous system with comprehensive comparison and is relatively simple to implement.

The basic circuit of the I&F neuron model is shown in Figure 1[6].
The basic circuit is in a dotted circle on the right of Figure 1. When the input current crosses the RC circuit, the voltage of the capacitor (black dot) is compared with the threshold voltage \( \theta \). If \( u(t) = \beta \) at time \( t^{(j)} \), an output spike \( \delta(t - t^{(j)}) \) will be generated. On the left side of Figure 1, the spike \( \delta(t - t^{(j)}) \) generated by the presynaptic neuron will be converted to the input spike \( \alpha(t - t^{(j)}) \) of the postsynaptic by a low-pass filter.

The basic circuit of the I&F model is composed of a capacitor \( C \) and a register \( R \) driven by the current \( I(t) \). The driven current consists of two parts. \( I(t) = I_R + I_c \). \( I_R \) is the current that crosses the register \( R \), and is calculated by Ohm’s Law \( I_R = u / R \). Here, \( u \) is the voltage across the register. \( I_c \) is the charge current of capacitor \( C \), which can be obtained from \( C = q / u \) and \( I_c = Cdu / dt \). The formula is as follows:

\[
I(t) = \frac{u(t)}{R} + C \frac{du}{dt}
\]

Equation (1) is multiplied by \( R \) and the introduction of the integration time constant \( \tau_m = RC \), and the standard equation can be obtained:

\[
\tau_m \frac{du}{dt} = -u(t) + RI(t)
\]

The paper supposes that an I&F neuron is stimulated by a series of spikes, and the value of the reset voltage is 0. This article assumes that a spike occurred at time \( t = t^{(j)} \). According to integral equation (2) and initial conditions \( u(t^{(0)}) = u_r = 0 \), we obtain the track of membrane voltage as follows:

\[
u(t) = RI_0[1 - \exp(-\frac{t - t^{(j)}}{\tau_m})]
\]

If the membrane voltage is less than \( \theta \), no further spikes will occur. On the contrary, if the membrane voltage is more than \( \theta \), the spike will occur. When the spike occurs, the membrane voltage will be reset to 0, and another integral process will begin.

**STDP Learning Rule**

The human brain has strong learning and memory features, and works based on the synaptic plasticity among neurons, which is the changing ability of the internal connection strength between presynaptic
and postsynaptic neuron is adaptive. Changing the synaptic weights corresponds to the neural network. Learning rules of the first two generations of ANN do not apply to SNN because of the temporal coding. The spike time dependent plasticity (STDP) rule, which evolved from the basis of the Hebb rule, is a rule of synaptic plasticity. It has a relationship with the order of firing time of two spikes, and its time-dependent theory is closer to the biological properties that are used widely in SNN. Therefore, the STDP learning algorithm is used in this paper.

The firing time difference of connecting spiking neural is used to modify the synaptic weights in STDP. As is shown in Figure 2, when the presynaptic neuron j fires before the postsynaptic neuron i, the connection strength between i and j will increase and the connection weights will increase too; Conversely, when the neuron i fires before neuron j, the connection strength between neurons will decrease and the connection weight will decrease.

![Figure 2. The schematic of the STDP learning rule.](image)

**MATLAB Implementation**

An SNN to curve fitting is constructed in this paper based on the spiking neurons and STDP learning rule. The network consists of a coding layer, processing layer and output layer. The information outside uses the coding layer as input, and is sent to the processing layer. The network will output results that are decoded by the output layer from the processing layer. The implementation process is as follows.

**Coding Layer**

The external information is transferred into a series of spikes in the coding layer. In this paper, a linear encoder is used for coding input data. As is shown in equation (4), $x$ represents the input data, $X$ represents the range of input data, and $T_{\text{max}}$ represents the maximum firing time of the spike which is defined as 20ms.

$$t = \frac{x}{X} T_{\text{max}}$$  \hspace{1cm} (4)

There are two curve fittings made in this paper, $y = x$ and $y = e^x$. For $y = x$, we obtain three points, ranging from 1 to 4. The actual input and designed output are shown in equation (5) and equation (6). For $y = e^x$, we take twenty points range from -1.2 ~ 6 in order to provide better curve fitting. The actual input and designed output are shown in equation (7) and equation (8).
\[ n = \frac{x \times 20}{5} \]  
\[ d(n) = \frac{x \times 20}{5} \]  
\[ n = \frac{(x + 2) \times 20}{9} \]  
\[ d(n) = \frac{e^{(x+2)} \times 20}{2981} \]

**Processing Layer**

The sampling time is 0.01ms of the neural network in the experiment, and the time window is 20ms, creating a total of 2,000 samples. The output spike signal of the coding layer is 0.1ms, which is as wide as the input of the network in this paper. Results proved that this approach is beneficial for voltage accumulation. When the voltage accumulation reaches 1.2 volts, neurons will generate a spike, and the firing time of the spike is regarded as the result for the input signal that achieves a process changing pulse sequence in the pulse voltage signal. Meanwhile, the network weights are constantly updated based on STDP learning rule. The training times of the network are 50 times, so the network weights are updated 50 times.

**Output Layer**

As the input and the desired output are coded to send into the network, the actual output should be decoded to correspond with the actual input. Equation (9) is the decoding function of \( y = x \), and equation (10) was the decoding function of \( y = e^x \).

\[ y(x) = \frac{u(n) \times 5}{20} \]  
\[ y(x) = \frac{u(n) \times 2981}{20} \]  

Where \( u(n) \) is the firing time corresponding to the input spike, and \( y(x) \) is the actual output value corresponding to the input. Finally, the actual input and actual output will be sent as an array.

**Experimental Results**

The relatively simple LI&F model reflects the biological characteristics used in this paper, and the synaptic plasticity law that synaptic weights change in index form of the STDP rule is followed. The functions of the SNN are realized by fitting \( y = x \) and \( y = e^x \) based on the MATLAB simulation platform.

The results of \( y=x \) are shown in Figure 3, and its actual input \( x=[1,3,4] \) are converted into n that corresponds to firing times of the spike by the information coding layer, where \( n=[4,12,16] \). Figure 3(a) shows the input spike of the network, which is obtained from coding the real input value and are also the output spikes of presynaptic neurons. Figure 3(b) shows changes of the membrane potential of postsynaptic neurons. Figure 3(c) shows contradictions between the actual output value and the desired output value after decoding; Figure 3(d) shows the error curve of a random output. As can be seen, the error will approach 0.01 after several training times.
The results of $y = ex$ are shown in Figure 4, and its actual input $x=[-1.2, 0, 0.5, 1.4, 2.1, 2.2, 2.3, 3.1, 3.5, 4, 4.2, 4.3, 4.4, 5, 5.1, 5.3, 5.4, 5.6, 6]$, a total of 20 values, are converted into $n$ firing times of spikes by the information coding layer. That is, $n=[1.7778, 4.4444, 5.5556, 7.5556, 8.4444, 9.1111, 9.3333, 9.5556, 11.3333, 12.2222, 13.3333, 13.7778, 14, 14.2222, 15.5556, 15.7778, 16.2222, 16.4444, 16.8889, 17.7778]$. Figure 4(a) shows the input spike of the network, which is obtained from coding the real input value and also are the output spikes of presynaptic neurons. Figure 4(b) shows changes in the membrane potential of postsynaptic neurons. Figure 4(c) shows contradictions between the actual output value and the desired output value after decoding. Figure 4(d) shows the error curve of a random output in the network. As can be seen, the error will approach 0.12 after several training times.
After comparing the actual output values with designed values at the end of the training, we find that the error tends to 0 and 0.3, using SNN for curving fitting of the linear function and the exponential function. It can be proved that curve fitting using SNN has the advantages of fast convergence and high accuracy.

Conclusion
Curve fitting is an important means of data processing. However, the traditional method of curve fitting has some limitations, such as polynomial fitting with low accuracy and cubic spline interpolation with high precision but long computation time. Therefore, SNN for curve fitting that is fast and accurate is used in this paper. First, the principle of the SNN is introduced in the paper. Second an SNN for curve fitting is established based on the MATLAB simulation platform. Finally, curve fitting for a linear function and exponential function is accomplished. After comparing the actual output values with designed values at the end of the training, we obtained the average error values of 0 and 0.3. The results show that the SNN has advantages of fast convergence and high accuracy on curve fitting.

Acknowledgement
This research was financially supported by the Fundamental Research Funds for the Central Universities, 2412016KJ016 and Thirteenth Five-Year Science and Technology Research Project of Education Department of Jilin Province, JJKH20170913KJ.

References

