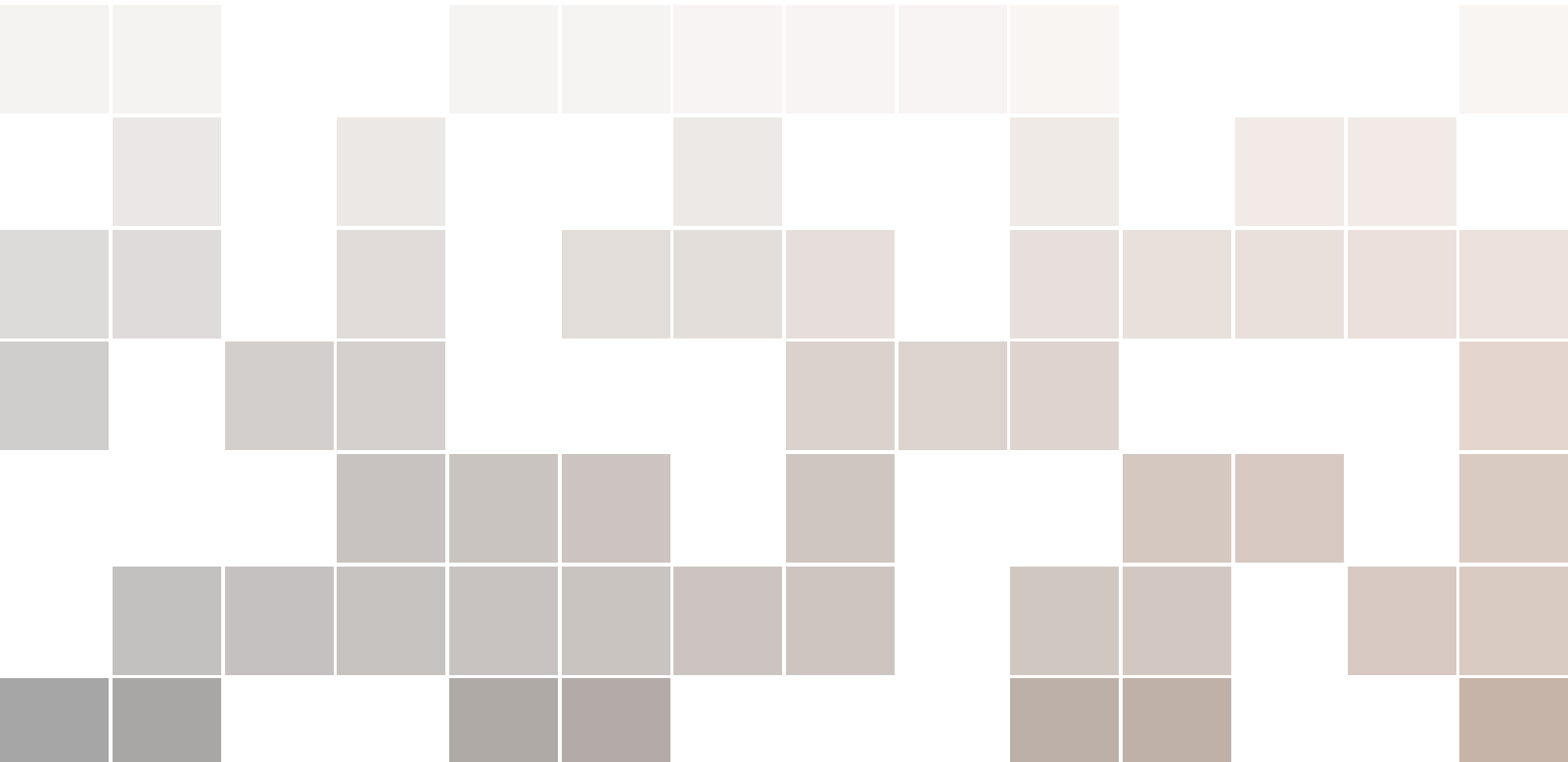




# Spiking Neural Network Handbook

(In Writing)

**Anonymous**







# Índice general

I	Part 1	
<b>1</b>	<b>Introduction</b>	<b>7</b>
<b>2</b>	<b>Neuron Dynamics</b>	<b>9</b>
<b>2.1</b>	<b>Biological Neuron</b>	<b>9</b>
<b>2.2</b>	<b>Neuron Model Abstract and Taxonomy</b>	<b>9</b>
2.2.1	Current-based Neuron	9
2.2.2	Conductance-based Neuron	10
<b>2.3</b>	<b>Neuron Model Examples</b>	<b>10</b>
2.3.1	Hodgkin-Huxley (HH) Model	10
2.3.2	Leaky Integrate-and-fire Model	11
2.3.3	Izhikevich Model	11
2.3.4	FitzHugh-Nagumo Model	11
2.3.5	Morris-Lecar Model	11
2.3.6	Hindmarsh-Rose Model	11
2.3.7	Cable theory	11
2.3.8	Perfect Integrate-and-fire	11
2.3.9	Adaptive Integrate-and-fire	11
2.3.10	Firing Rate Model	11
2.3.11	Discussion	11
<b>3</b>	<b>Synapse Dynamics</b>	<b>13</b>
<b>3.1</b>	<b>Biological Synapse</b>	<b>13</b>
<b>3.2</b>	<b>Synapse Abstract and Taxonomy</b>	<b>13</b>
3.2.1	Current-based Synapse	13

3.2.2	Conductance-based Synapse	13
3.2.3	Chemical Synapse	13
<b>3.3</b>	<b>Discussion</b>	<b>13</b>
<b>4</b>	<b>Training Algorithms</b>	<b>15</b>
<b>4.1</b>	<b>Unsupervior Learning</b>	<b>15</b>
4.1.1	Spike-timing-dependent plasticity (STDP)	15
4.1.2	Growing Spiking Neural Networks	15
4.1.3	Artola, Bröcher, Singer (ABS) rule	15
4.1.4	Bienenstock, Cooper, Munro (BCM) rule	15
4.1.5	Relationship between BCM and STDP rules	15
<b>4.2</b>	<b>Supervised Learning</b>	<b>15</b>
4.2.1	STDP-based Methods	15
4.2.2	Spike-Timing Dependent Backpropagation (STDBP)	15
4.2.3	Liquid State Machine (LSM) and Readout Training	15
4.2.4	SpikeProp	15
4.2.5	ReSuMe	15
4.2.6	SuperSpike	16
4.2.7	SPAN (Mohammed et al., 2012)	16
4.2.8	Remote Supervised Method (ReSuMe)	16
4.2.9	FreqProp	16
4.2.10	Local error-driven associative biologically realistic algorithm (LEABRA)	16
4.2.11	Supervised Hebbian Learning	16
<b>4.3</b>	<b>Reinforcement Learning</b>	<b>16</b>
4.3.1	Spiking Actor-Critic method	16
4.3.2	STDP-based Methods	16
<b>4.4</b>	<b>Convert Trandictional ANN to SNN</b>	<b>16</b>
<b>5</b>	<b>Network Architecture Investigation</b>	<b>17</b>
<b>5.1</b>	<b>Liquid Neural Network</b>	<b>17</b>
<b>5.2</b>	<b>Feedforward Neural Network</b>	<b>17</b>
<b>5.3</b>	<b>Recurrent Neural Network</b>	<b>17</b>
<b>5.4</b>	<b>Synfire Chain</b>	<b>17</b>
<b>5.5</b>	<b>Reservoir computing</b>	<b>17</b>
	<b>Indexes</b>	<b>19</b>

# Part 1

<b>1</b>	<b>Introduction .....</b>	<b>7</b>
<b>2</b>	<b>Neuron Dynamics .....</b>	<b>9</b>
2.1	Biological Neuron	
2.2	Neuron Model Abstract and Taxonomy	
2.3	Neuron Model Examples	
<b>3</b>	<b>Synapse Dynamics .....</b>	<b>13</b>
3.1	Biological Synapse	
3.2	Synapse Abstract and Taxonomy	
3.3	Discussion	
<b>4</b>	<b>Training Algorithms .....</b>	<b>15</b>
4.1	Unsupervised Learning	
4.2	Supervised Learning	
4.3	Reinforcement Learning	
4.4	Convert Traditional ANN to SNN	
<b>5</b>	<b>Network Architecture Investigation ...</b>	<b>17</b>
5.1	Liquid Neural Network	
5.2	Feedforward Neural Network	
5.3	Recurrent Neural Network	
5.4	Synfire Chain	
5.5	Reservoir computing	
	<b>Indexes .....</b>	<b>19</b>





## 1. Introduction







## 2. Neuron Dynamics

Spiking neural network's neuron model describe how the membrane potential of a neuron change over the time. In this chapter, we dive into the biological neuron's dynamics and exhibit several examples neuron models in the spiking neural network.

### 2.1 Biological Neuron

### 2.2 Neuron Model Abstract and Taxonomy

In spiking neural network, Equation 2.2.1 to Equation 2.2.4 provide a highly abstract description for modeling a biological neuron. In these equation,  $\mathbf{x}_t$  is the state vector at time  $t$ ,  $\Delta\mathbf{x}_t$  is the state variation at time  $t$ ,  $\mathbf{u}_t$  is the input vector at time  $t$ ,  $V$  is the membrane potential of the neuron,  $\mathbf{y}_t$  is the neuron's output at time  $t$ , and  $V_{out}$  is the output voltage that will be sent to the synapses that departure from this neuron.  $(\mathbf{x}_t)_i$  is the  $i$ -th element of the state vector.

$$\Delta\mathbf{x}_t = f(\mathbf{x}_t, \mathbf{u}_t) \quad (2.2.1)$$

$$\mathbf{y}_t = g(\mathbf{x}_t, \mathbf{u}_t) \quad (2.2.2)$$

$$s.t., (\exists i \in [0, |\mathbf{x}_t|])(\mathbf{x}_t)_i = V \quad (2.2.3)$$

$$(\exists i \in [0, |\mathbf{y}|])_i \mathbf{y} = V_{out} \quad (2.2.4)$$

These equations are in the form of a **state-space model (SMM)**.

Equation 2.2.3 indicates that, a neuron should maintain a membrane potential  $V$ , and Equation 2.2.4 indicates that, the neuron's output should contains a voltage  $V_{out}$ .

#### 2.2.1 Current-based Neuron

In a current-based neuron model, the synaptic input is represented as an injected current directly added to the membrane potential equation. we have the input  $\mathbf{u}_t$  contains the current  $I_t$ , and  $\mathbf{x}_t$  contains the membrane potential  $V_t$ .

We will later to see that a current-based LIF neuron (Section 2.3.2) can hold a form of Equation 2.2.5.

$$\dot{V}_t = \frac{1}{\tau_m}(-(V_t - V_{rest}) + I_t) = f_{CurrentLIF_1}(V_t, \mathbf{u}_t = [I_t]) \quad (2.2.5)$$

### 2.2.2 Conductance-based Neuron

In a conductance-based neuron model, the synaptic input is modeled by changing the conductance of the membrane, which then affects the current flow.

In the state space representation, the input  $\mathbf{u}_t$  is the synapse conductance. let  $\mathbf{u}_t = g_{syn}(t)$ ,  $\mathbf{x}_t = V_t$ , we will see that in conductance-based LIF neuron model, it hold Equation 2.2.6

$$\dot{V}_t = \frac{1}{\tau_m}(-(V_t - V_{rest}) + g_{syn}(t)(E_{syn} - V_t)) = f_{ConductanceLIF_1}(V_t, \mathbf{u}_t = g_{syn}(t)) \quad (2.2.6)$$

## 2.3 Neuron Model Examples

### 2.3.1 Hodgkin-Huxley (HH) Model

The Hodgkin-Huxley (HH) model is a conductance-based model , which can be utilize to accurately reproduce the bio-neuron's dynamics. Its form is shown in Equation 2.3.1 to Equation 2.3.5. Combine all these equations, we get Equation 2.3.6.

$$I = C_m \frac{dV_m}{dt} + I_i \quad (2.3.1)$$

$$I_i = I_{Na} + I_K + I_l \quad (2.3.2)$$

$$I_{Na} = g_{Na}(V_m - V_{Na}) \quad (2.3.3)$$

$$I_K = g_K(V_m - V_K) \quad (2.3.4)$$

$$I_l = \bar{g}_l(V_m - V_l) \quad (2.3.5)$$

$$I_l = C_m \frac{dV_m}{dt} + g_{Na}(V_m - V_{Na}) + g_K(V_m - V_K) + \bar{g}_l(V_m - V_l) \quad (2.3.6)$$

Ion channel function  $g$ . are function respect to time  $t$  and membrane potential  $V$ . Specifically, Equation 2.3.7 is held.

$$g_{Na} = \bar{g}_{Na}m^3h \quad g_K = \bar{g}_Kn^4 \quad g_l = \bar{g}_l \quad (2.3.7)$$

Combine Equation 2.3.1 to Equation 2.3.7, we get Equation 2.3.8

$$I_l = C_m \frac{dV_m}{dt} + \bar{g}_{Na}m^3h(V_m - V_{Na}) + \bar{g}_Kn^4(V_m - V_K) + \bar{g}_l(V_m - V_l) \quad (2.3.8)$$

$\frac{d}{dt} = \alpha \cdot (V_m)(1 - \cdot) - \beta \cdot (V_m) \cdot$  is held. Where  $\cdot$  is a placeholder for  $m, n$  and  $h$ . As such, Equation 2.3.9 to Equation 2.3.11 are held.

$$\frac{dn}{dt} = \alpha_n(V_m)(1 - n) - \beta_n(V_m)n \quad (2.3.9)$$

$$\frac{dm}{dt} = \alpha_m(V_m)(1 - m) - \beta_m(V_m)m \quad (2.3.10)$$

$$\frac{dh}{dt} = \alpha_h(V_m)(1 - h) - \beta_h(V_m)h \quad (2.3.11)$$

$$(2.3.12)$$

From experiment, we have Equation 2.3.13 to Equation 2.3.18.

$$\alpha_n(V_m) = \frac{0,01(10 - V)}{\exp(\frac{10-V}{10}) - 1} \quad (2.3.13)$$

$$\alpha_m(V_m) = \frac{0,1(25 - V)}{\exp(\frac{25-V}{10}) - 1} \quad (2.3.14)$$

$$\alpha_h(V_m) = 0,07\exp(-\frac{V}{20}) \quad (2.3.15)$$

$$\beta_n(V_m) = 0,125\exp(-\frac{V}{80}) \quad (2.3.16)$$

$$\beta_m(V_m) = 4\exp(-\frac{V}{18}) \quad (2.3.17)$$

$$\beta_h(V_m) = \frac{1}{\exp(\frac{30-V}{10}) + 1} \quad (2.3.18)$$

Hodgkin-Huxley could be seen as a current-based neuron model, which may represent by space state model, with  $\mathbf{x}_t = [V_t, m_t, h_t, n_t]$ , and  $\mathbf{u}_t = I_t$ .

### 2.3.2 Leaky Integrate-and-fire Model

Leaky Integrate-and-fire model is a computational effective model, in which a threshold is set, when membrane potential cross the threshold, the neuro emit a spike. In implementation, we may set the voltage output at time  $t$ ,  $V_{out}$  be 1 mV. The scale problem has the potential to be solved by automatically adjusting the synapses' weights in training. A LIF neuron  $i$ 's output form a **spike train**  $S_i(t) = \sum_m \delta(t - t_m)$ .

$$\dot{V}_t = \frac{1}{\tau_m} (-(V_t - V_{rest}) + g_{syn}(t)(E_{syn} - V_t)) \quad (2.3.19)$$

### 2.3.3 Izhikevich Model

### 2.3.4 FitzHugh-Nagumo Model

### 2.3.5 Morris-Lecar Model

### 2.3.6 Hindmarsh-Rose Model

### 2.3.7 Cable theory

### 2.3.8 Perfect Integrate-and-fire

### 2.3.9 Adaptive Integrate-and-fire

### 2.3.10 Fring Rate Model

### 2.3.11 Discussion

**Spike Representation** You may note that, although different neuron models have different dynamics and spike representations, they can still communicate with each other through synapses. Some neuron models require current input, while others do not. Nonetheless, spikes can transmit over synapses and cause current variations across them by utilizing a common output element  $V_{out}$ .

One challenge that may arise is the normalization of spike representations. To address this issue, one approach is to standardize the spike events by converting all spike representations into binary form. Another approach can involve adjusting the synaptic weights. Although we may face scaling issues with different types of spike representations, these can be mitigated during training by appropriately adjusting the synapse weights.





## 3. Synapse Dynamics

### 3.1 Biological Synapse

### 3.2 Synapse Abstract and Taxonomy

#### 3.2.1 Current-based Synapse

#### 3.2.2 Conductance-based Synapse

#### 3.2.3 Chemical Synapse

Current-based Synapse

Conductance-based Synapse

### 3.3 Discussion





## 4. Training Algorithms

### 4.1 Unsupervised Learning

#### 4.1.1 Spike-timing-dependent plasticity (STDP)

#### 4.1.2 Growing Spiking Neural Networks

#### 4.1.3 Artola, Bröcher, Singer (ABS) rule

#### 4.1.4 Bienenstock, Cooper, Munro (BCM) rule

#### 4.1.5 Relationship between BCM and STDP rules

### 4.2 Supervised Learning

#### 4.2.1 STDP-based Methods

Supervised STDP (SSTDP)

Spike-Timing-Dependent Plasticity (STDP) with Supervision

#### 4.2.2 Spike-Timing Dependent Backpropagation (STDBP)

#### 4.2.3 Liquid State Machine (LSM) and Readout Training

#### 4.2.4 SpikeProp

**Extension** (McKennoch et al., 2006; Booi and tat Nguyen, 2005; Shrestha and Song, 2015; de Montigny and Mâsse, 2016; Banerjee, 2016; Shrestha and Song, 2017).

spike timing based methods is that they cannot learn starting from a quiescent state of no spiking. Bohte (2011)

Huh and Sejnowski (2017)

#### 4.2.5 ReSuMe

**Related Work** (Sporea and Grüning, 2013) Pfister et al. (2006) Gardner et al. (2015) Fremaux et al. (2010)

- 4.2.6 SuperSpike
- 4.2.7 SPAN (Mohammed et al., 2012)
- 4.2.8 Remote Supervised Method (ReSuMe)
- 4.2.9 FreqProp
- 4.2.10 Local error-driven associative biologically realistic algorithm (LEABRA)
- 4.2.11 Supervised Hebbian Learning
- 4.3 Reinforcement Learning
  - 4.3.1 Spiking Actor-Critic method
  - 4.3.2 STDP-based Methods
- 4.4 Convert Trandictional ANN to SNN





## 5. Network Architecture Investigation

### 5.1 Liquid Neural Network

### 5.2 Feedforward Neural Network

### 5.3 Recurrent Neural Network

### 5.4 Synfire Chain

### 5.5 Reservoir computing





## Índice alfabético

### A

Adaptive Integrate-and-fire . . . . . 11

### B

Biological Neuron . . . . . 9

Biological Synapse and Taxonomy . . . . . 13

### C

Cable theory . . . . . 11

Conductance-based Neuron . . . . . 10

Current-based Neuron . . . . . 9

### D

Discussion (Neuron Dynamics) . . . . . 11

### F

Firing Rate Model . . . . . 11

FitzHugh-Nagumo Model . . . . . 11

### H

Hindmarsh-Rose Model . . . . . 11

Hodgkin-Huxley (HH) Model . . . . . 10

### I

Izhikevich Model . . . . . 11

### L

Leaky Integrate-and-fire Model . . . . . 11

### M

Morris-Lecar Model . . . . . 11

### N

Neuron Model Abstract and Taxonomy . . . . . 9

Neuron Model Examples . . . . . 10

### P

Perfect Integrate-and-fire . . . . . 11