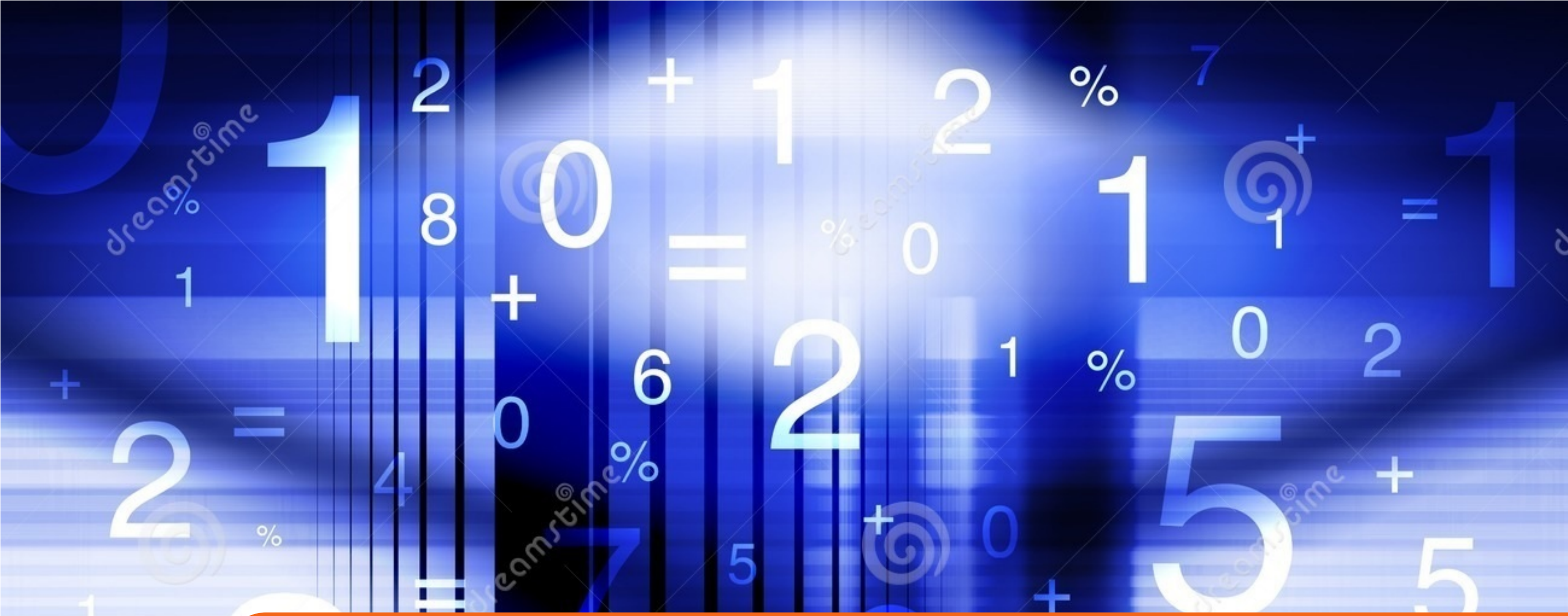
Spiking Neural Network Handbook (In Writing)

Anynomous

Índice general

Part 1

1. Introduction



2. Neuron Dynamics

The neuron models of spiking neural network model how the membrance potential of a neuron change over the time. Three classical models include `Hodgkin-Huxley (HH) model`, `Leaky integrate-and-fire model`, and `Zhikevich model`.

2.1 Biological Neuron

For all kinds of neuron dynamics, we may use a space state model $\dot{\mathbf{x}}_{\mathbf{t}} := f_{\mathscr{P}}(\mathbf{x}_t, \mathbf{u}_t)$ to represent it. In which, $\dot{\mathbf{x}}$ is the neuron's states at time t, $dotx$ is the variation of states at time t, and $\mathbf{u}_t$ is the input at time t.

2.2 Neuron Model Abstract and Taxonomy

In spiking neural network, we may use Equation 2.2.1 to Equation 2.2.3 to model a biological neuron. Where $\mathbf{x}_t$ is the state vector at time t, Δx 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, and V_{out} is the output voltage that will be sent to the synapses that depature from this neuron. $(\mathbf{x}_{t_i})$ is the i-th element of the state vector.

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

$$\Delta \mathbf{V}_{out} = g(\mathbf{x}_t + \Delta \mathbf{x}_t, \mathbf{u}_t) \tag{2.2.2}$$

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

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 u_t is the current input, and $\mathbf{x}$ contains the membrance potential.
We will later to see that the current-based LIF neuron **??** can hold a form of Equation **??** similar to state space representation.

$$\dot{V}_t = \frac{1}{\tau_m}(-(V_t - V_{rest}) + I_t) = f_{\mathscr{P}}(V_t, I_t) \tag{2.2.4}$$

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 2.2.5

$$\dot{V}_t = \frac{1}{\tau_m}(-(V_t - V_{rest}) + g_s yn(t)(E_{syn} - V_t)) = f_{\mathscr{P}}(V_t, g_{syn}(t)) \tag{2.2.5}$$

$$\dot{x} = f_P(x_t, u_t) \tag{2.2.6}$$

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.

$$I = C_m \frac{dV_m}{dt} + I_i \tag{2.3.1}$$
$$I_i = I_{Na} + I_K + I_l \tag{2.3.2}$$
$$I_{Na} = g_{Na}(V_m - V_{Na}) \tag{2.3.3}$$
$$I_K = g_K(V_m - V_K) \tag{2.3.4}$$
$$I_l = \bar{g}_l(V_m - V_l) \tag{2.3.5}$$

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

Ion channel function $g_\cdot$ are function respect to time t and membrance potential V. Specifically, Equation 2.3.7 is held.

$$g_{Na} = \bar{g}_{Na} m^3 h \qquad g_K = \bar{g}_K n^4 \qquad g_l = \bar{g}_l \tag{2.3.7}$$

Combine Equation 2.3.1 to Equation 2.3.7, we get Equation 2.3.8

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

$\frac{d\cdot}{dt} = \alpha_\cdot(V_m)(1 - \cdot) - \beta_\cdot(V_m)\cdot$ 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.

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

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

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

$$\tag{2.3.12}$$

From experiment, we have Equation 2.3.13 to Equation 2.3.18.

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

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

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

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

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

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

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 membrance potential cross the threshold, the neuro emit a spike. Usually the spike could be represented by value 1.
It hold the form of Equation 2.3.19

$$C_m \frac{dV_m(t)}{dt} = I(t) - \frac{V_m(t)}{R_m} \tag{2.3.19}$$

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, though different neuron model hold different dynamics, and spike representation. Some need current input, and some not require this kind of input.

Nontheless, they can communicated with each other though synapses. Spikes can transmit over synapses, and cause current variation over the synapses. A problem may be the normalization of spike representations. To tackle with this problem, one approach can be standardize the spike events, making all spike representation into binary representation. Another weight can be dependent on the synapses. Though we may face scale problems, in different kinds of spike representation, by adjusting Synapse weights in trainning, this problem can be mitigated.

3. Synapse Dynamics

3.1 Biological Synapse

3.2 Synapse Abstract and Taxonomy

Similar to neuron dynamics, the synapse dynamics could be modeled by state space model, and the current I is included in the state $\mathbf{x}$.

3.2.1 Current-based Synapse

In current-based synapses, the postsynaptic effect is modeled as an instantaneous current added to the postsynaptic neuron. Example: Synaptic current as an exponential decay

$$I_{syn}(t) = I_{peak} e^{-(t-t_{spike})/\tau_s} \tag{3.2.1}$$

I_{peak}: peak current. t_{spike}: the time of the presynaptic spike. τ_s: the time constant of the synaptic current.
In the context of synaptic models, the V in the synaptic current equation generally refers to the membrane potential of the postsynaptic neuron

3.2.2 Conductance-based Synapse

E.g., Alpha function synapse: $g_{syn}(t) = g_{max} \frac{t-t_{spike}}{\tau} e^{-\frac{t-t_{spike}}{\tau}}$ The synaptic current then becomes: $I_{syn}(t) = g_{syn}(t)(E_{syn} - V(t))$

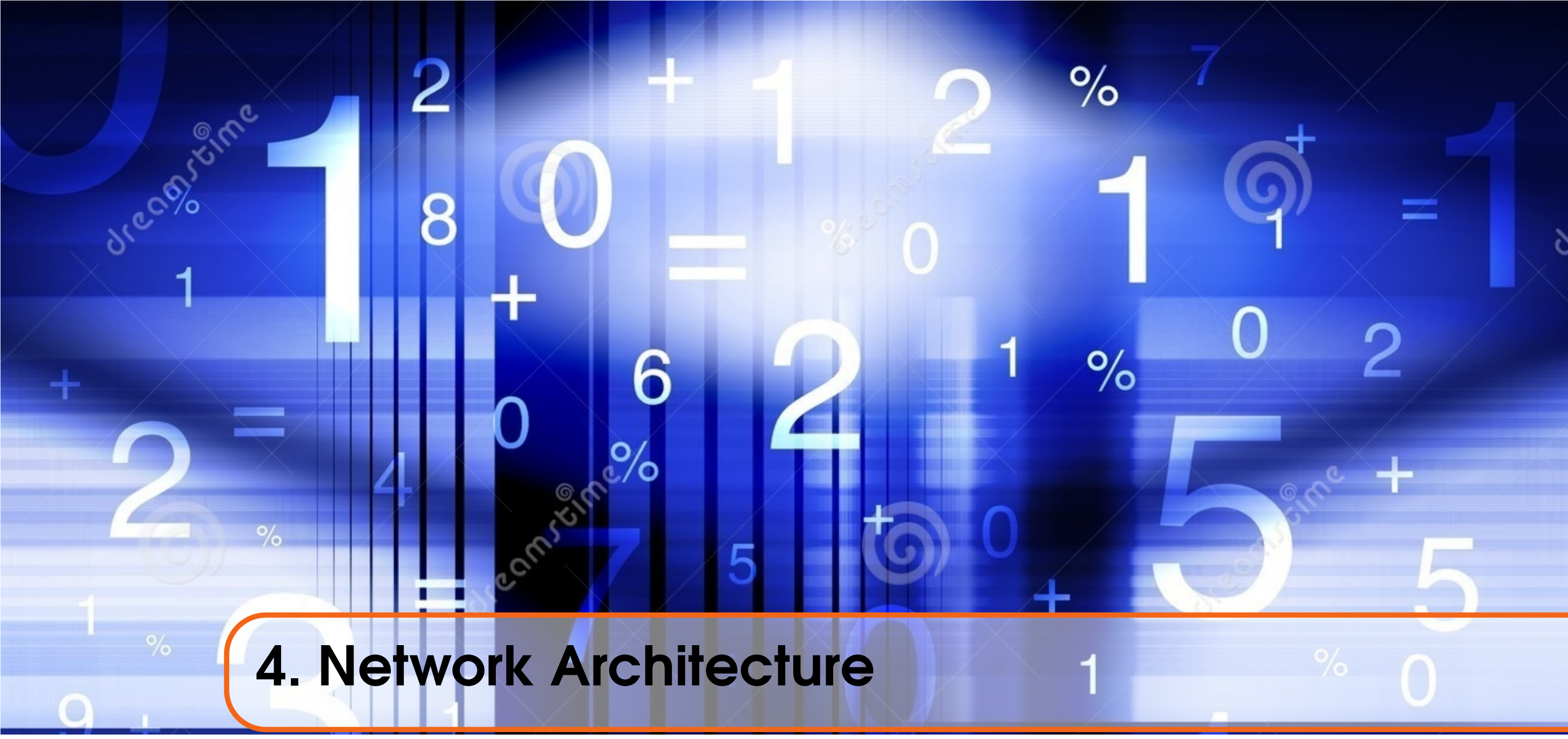
3.2.3 Chemical Synapse

Chemical synapses can be modeled using either current-based or conductance-based approaches, but these are modeling choices rather than fundamentally different categories.
Current-based Synapse
Modeling Approach: In a current-based synapse model, the effect of neurotransmitter release is represented as an injected current directly added to the postsynaptic neuron's membrane potential.
Conductance-based Synapse

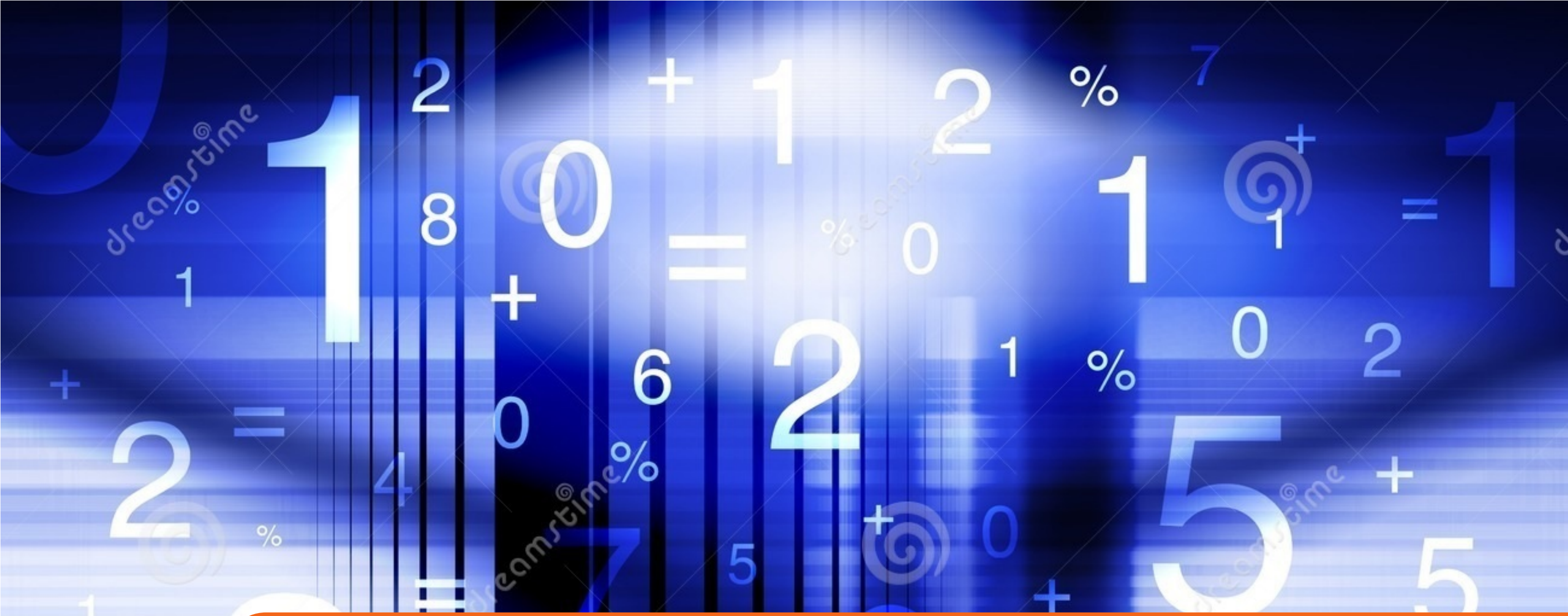
Modeling Approach: In a conductance-based synapse model, the effect of neurotransmitter release is modeled by changing the synaptic conductance, which then affects the current flow based on the difference between the membrane potential and the synaptic reversal potential.

3.3 Discussion

In a summary, all kind of neuron should maintains a state of membrance V, its variation can in a current-based method or a conductance-based method. The variation of V can make the current changed in synapse. And current I is a nessary state maintained by synapses no matter in what kind of methods. Though in a conductance-based method, we can retrieve the synapse I from equations.



4. Network Architecture



5. Trainning Algorithms

Trainning a spike neural network remain chanllenging. In this section, we will review the exist methods.

5.1 Unsupervior Learning

5.1.1 Spike-timing-dependent plasticity (STDP)

STDP can be devised from information maximizing principles (Bohte and Mozer, 2007; Toyoizumi et al., 2005).

5.1.2 Growing Spiking Neural Networks

5.1.3 Artola, Bröcher, Singer (ABS) rule

5.1.4 Bienenstock, Cooper, Munro (BCM) rule

5.1.5 Relationship between BCM and STDP rules

5.2 Supervised Learning

For supervised learning, there is some methods, including

5.2.1 STDP-based Methods

Supervised STDP (SSTDP)

Spike-Timing-Dependent Plasticity (STDP) with Supervision

5.2.2 Spike-Timing Dependent Backpropagation (STDBP)

5.2.3 Liquid State Machine (LSM) and Readout Training

5.2.4 SpikeProp

Extension (McKennoch et al., 2006; Booij 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)

5.2.5 ReSuMe

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

5.2.6 SuperSpike

SuperSpike [**super-spike**] is a supervisor learning algorithm dedicated to deterministic LIF neurons. In the original paper, the authors address three problems in trainning a SNN:

- Outputs' complexity due to the spatiotemperal nature: the output are spatiotemperal patterns, we cannot independently consider them only on temperal or spatial.
- Non-differentiable at spike time: trandictional gradient-based optimization methods face chanllenging in the trainning of SNN.
- Problems in analytically treat self-memory.

credit assignment in hidden layers is problematic:

- Lacks auto-differentiation tools.
- Chanllenging in devising biologically plausible weight update strategies.

In a similar vein, the Chronotron (Florian, 2012) learns precisely timed output spikes by minimizing the Victor-Pupura distance (Victor and Purpura, 1997) to a given target output spike train.
SuperSpike provide a supervised learning approach for trainning the SNN. This appraoch can tackle with networks that have hidden units.
Approaches approximate the partial derivative of the hidden unit output by $f(S_{pre}, g(V_{post}))$.
Let $\hat{S}_i$ be the target spike train of neuron i. The cost model for optimization that make $\hat{S}_i$ approach the real S_i hold the form: $L = \frac{1}{2}\int_{-\infty}^{t} ds[(\alpha * \hat{S}_i - \alpha * S_i)(s)]^2$.
α is a normalized smooth temporal convolution kernel. The original SuperSPike use *double exponential causal kernel*.

$$\partial L/\partial w_{ij} = -\int_{-\infty}^{t} ds[(\alpha * \hat{S}_i - \alpha * S_i)(s)](\alpha * \frac{\partial S_i}{\partial w_{ij}})(s)$$

.
Some existing methods for tackling the term $\frac{\partial S_i}{\partial w_{ij}}$: (1) making derivation directly to the membrance voltage, (2) introducing noisy which render the likelihood of $\langle S_i \rangle$ a smooth function of the membrance potential.
The superspike convert calculation of $\frac{\partial S_i}{\partial w_{ij}} \to \sigma'(U_i)\frac{\partial U_i}{\partial w_{ij}}$. In which U_i is the membrance voltage.
Original superspike choose $\sigma(U)$ be the negative side of a fast sigmoid. This function is objective to increase steeply and peak at the spiking threshold. Other monotonic functions may also work.
For current-based LIF models the membrane potential $U_i(t)$ can be written in integral form as a spike response model (SRM0 (Gerstner et al., 2014)): $U_i(t) = \sum_j w_{ij}(\varepsilon * S_j(t)) + (\eta * S_i(t))$
ε corresponds to the postsynaptic potential (PSP) shape, η captures spike dynamics and reset.
The existence of term $(\eta * S_i(t))$ make us different to calculate U_i's derivation with respect of wij.
In the condition that fires rate are low (it is psychological pausible), U_i has low correlation to this term.
So they further remove the second term. Now $U_i(t) \approx (\varepsilon * S_j(t))$.
The gradient calculation for a weight become:

$$\frac{\partial w_{ij}}{\partial t} = r\int_{-\infty}^{t} ds e_i(s)\alpha * (\sigma'(U_i(s))(\varepsilon * S_j)(s))$$

.
r is the learning rate, $e_i(s) \equiv \alpha * (\hat{S}_i - S_i)$, $\lambda_{ij} = \alpha * (\sigma'(U_i(s))(\varepsilon * S_j)(s))$ is the eligibility trace.
The form above is also known as *non-vanishing surrogate gradient*.

Neuron Model

$$\tau^{mem}\frac{dU_i}{dt} = (U^{rest} - U_i) + I_i^{syn}(t)$$

Synapse Current Evolution

$$\frac{d}{dt}I_i^{syn}(t) = -\frac{I_i^{syn}(t)}{\tau^{syn}} + \sum_{j\in pre} w_{ij}S_j(t)$$

Distal Reward Problem Eligibility trace
Causal Convolution
Nonlinear Hebbian term detects coincidences between presynaptic activity and postsynaptic depolarization. Hebbian three factor rule These spatiotemporal coincidences at the single synapse wij are then stored transiently by the temporal convolution with the causal kernel α. This step can be interpreted as a synaptic eligibility trace, which in neurobiology could for instance be implemented as a calcium transient or a related signaling cascade.

$$\frac{\partial w_{ij}}{\partial t} = r_{ij}\int_{t_k}^{t_{k+1}} e_i(s)\alpha * (\sigma'(U_i(s))(\varepsilon * S_j)(s))ds$$

.
ε is a double exponential filter. The original SuperSpike use two single exponential filter to realize it. In each time step, it firstly integrate the single exponential trace by $\frac{dz_j}{dt} = -\frac{z_j}{\tau_{rise}} + S_j(t)$. It then fed into a second exponential filter array $\tau_{decay}\frac{\tilde{z}_j}{dt} = -\tilde{z}_j + z_j$.
$\tilde{z}_j(t) \equiv (\varepsilon * S_j)(t)$.
$\sigma'(U_i) = (1 + |h_i|^2)$ with $h_i \equiv \beta(U_i - \ell)$, where ℓ is the neuronal firing threshold.
Error signals include output error signal and feedback error signal. The formmer is the error between output spike train and predicted spike train. Feedback signals are signals that derived from the output error signal.
$\alpha \propto \varepsilon$. For output signal $e_i = -\alpha * (\tilde{S}_i - S_i)$. For feedback signals (1) Symmetric Feedback: $e_i = \sum_k w_{ki}e_k$, (2) Random Feedbck: $e_i = \sum_k b_{ki}e_i$, where $b_{ki} \sim N(0,1)$, (3) Uniform Feedback: $e_i = \sum_k e_k$.
For weight update it use a separate variable m_{ij} in a specific chunk size t_b. $m_{ij} \leftarrow m_{ij} + g_{ij}$, $g_{ij}(t) = e(t)\lambda_{ij}(t)$. At the end of each t_b, $w_{ij} \leftarrow w_{ij} + r_{ij}m_{ij}$, where r_{ij} it the learning rates. They ensure $-0{,}1 < w_{ij} < 0{,}1$.
In some experiment, regularity term may be added in the the weight learning rule.

$$\frac{\partial w_{ij}^{hid}}{\partial t} = r_{ij}\int_{t_k}^{t_{k+1}} e_i(s)(\alpha * (\sigma'(U_i(s))(\varepsilon * S_j)(s)) - \rho w_{ij}e_i(s)z_i^4)ds$$

.
Regularization strength ρ was chosen to be the square of error.
$\frac{z_i}{dt} = -\frac{z_i}{\tau_{het}} + S_i(t)$.
Pprobabilistic escape rate model (Pfister et al. (2006))
Victor-Pupura distance-based learning (Victor and Purpura, 1997)
Convergence properties of rules that reduce the van Rossum distance by gradient descent. (Gardner and Grüning (2016) and Albers et al. (2016))
Learning algorithm Proposed by Memmesheimer et al. (2014)
Sequence Learning Problem as a Variational Learning Problem
Combining adaptive control theory with heterogeneous neurons. (Gilra and Gerstner, 2017)
Tempotron (Gütig and Sompolinsky, 2006; Gütig, 2016) can be derived as a gradient-based approach (Urbanczik and Senn, 2009).
It is chanllenging in calcultion of $\frac{\partial S_i(t)}{\partial w_{ij}}$, where $S_i(t) = \sum_k \delta(t - t_i^k)$. $S_i(t)$ is the spike train of neuron i.

5.2.7 SPAN (Mohemmed et al., 2012)

5.2.8 Remote Supervised Method (ReSuMe)

5.2.9 FreqProp

5.2.10 Local error-driven associative biologically realistic algorithm (LEABRA)

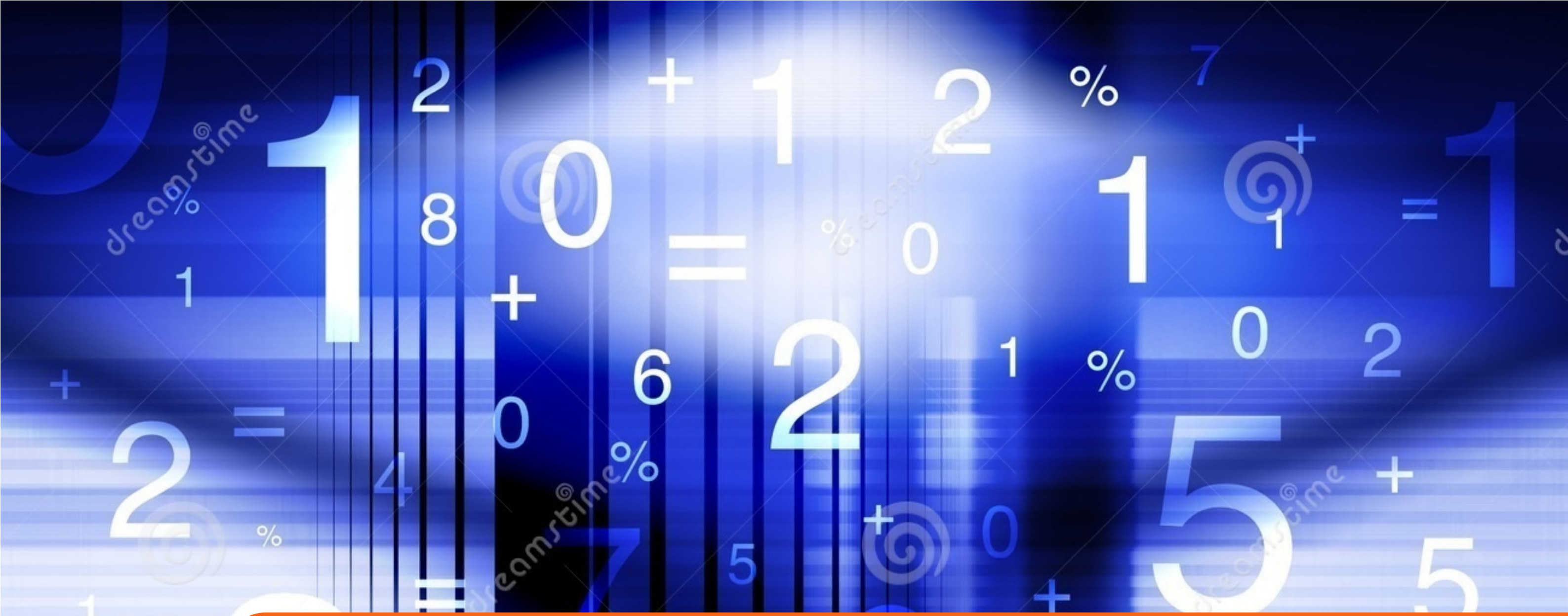
5.2.11 Supervised Hebbian Learning

5.3 Reinforcement Learning

5.3.1 Spiking Actor-Critic method

5.3.2 STDP-based Methods

5.4 Convert Trandictional ANN to SNN



Índice alfabético