Gradient Descent for Spiking Neural Networks

Dongsung Huh
Salk Institute
La Jolla, CA 92037
huh@snl.salk.edu

Terrence J. Sejnowski
Salk Institute
La Jolla, CA 92037
terry@salk.edu

Abstract

Much of studies on neural computation are based on network models of static neurons that produce analog output, despite the fact that information processing in the brain is predominantly carried out by dynamic neurons that produce discrete pulses called spikes. Research in spike-based computation has been impeded by the lack of efficient supervised learning algorithm for spiking networks. Here, we present a gradient descent method for optimizing spiking network models by introducing a differentiable formulation of spiking networks and deriving the exact gradient calculation. For demonstration, we trained recurrent spiking networks on two dynamic tasks: one that requires optimizing fast ($\approx$ millisecond) spike-based interactions for efficient encoding of information, and a delayed memory XOR task over extended duration ($\approx$ second). The results show that our method indeed optimizes the spiking network dynamics on the time scale of individual spikes as well as behavioral time scales. In conclusion, our result offers a general purpose supervised learning algorithm for spiking neural networks, thus advancing further investigations on spike-based computation.

1 Introduction

The brain executes highly dynamic computation over multiple time-scales: Individual neurons exhibit spiking dynamics of millisecond resolution and the hierarchy of recurrent network connections produces slower dynamics on the order of seconds to minutes. How the brain organizes spiking dynamics to form the basis for computation is a central problem in neuroscience. Nonetheless, most analysis and modeling of neural computation assume static neural models that produce analog output [1, 2]. These simplified models are compatible with the advanced tools from the deep learning field that can efficiently optimize large scale network models to perform complex computational tasks [3]. However, such models fail to describe the fast dynamics of spike-based computation in the brain.

The main difficulty in optimizing spiking neural networks stems from the discrete, all-or-none nature of spikes: A spiking neuron generates a brief spike output when its membrane voltage crosses the threshold, and silent at other times. This non-differentiable behavior is incompatible with the standard, gradient-based supervised learning methods. Therefore, learning algorithms for spiking neural networks explored various ways to circumvent the non-differentiability problem.

SpikeProp [4] considered spike times of neurons as state variables and used the differentiable relationship between the input and the output spike times to minimize the difference between the actual and the desired output spike times. However, the creation and deletion of spikes is non-differentiable, so the number of spikes must be pre-specified. Memmesheimer et al [5] considered the problem of generating spikes at desired times and remaining silent at other times as a binary classification problem and applied the perceptron learning algorithm (See also [6].) Pfister et al [7] considered stochastic spiking neurons and maximized the smooth likelihood function of the neuron spiking at desired times. More biologically inspired methods based on spike-time-dependent-plasticity (STDP) have also been proposed [8, 9, 10, 11]. All of these methods, however, require

target spiking activity of individual neurons at desired times, which significantly limit their range of applicability.

Alternative methods have also been proposed: Instead of directly optimizing a spiking network, these methods optimize a network of static, analog neurons and replicate the optimized solution with a spiking network. Hunsberger and Eliasmith [12] used analog units that closely approximated the firing rate of individual spiking neurons. Instead of replicating individual neuron's dynamics, Abbott et al [13] proposed replicating the entire network dynamics using recently developed methods from predictive coding [14]. Although these approaches can be applied to a wider range of problems, the trained spiking network can only mimic the solution of the analog, rate-based networks, rather than exploring the larger space of spike-time based solutions.

In this paper, we introduce a novel, differentiable formulation of spiking neural networks and derive the exact gradient calculation for gradient based optimization. This method optimizes the network dynamics on the time scale of individual spikes for general supervised learning problems.

2 Methods

Section 2.1 reformulates the synaptic current dynamics model in a differentiable form. Section 2.2 introduces other components of the spiking network, and section 2.3 describes the gradient calculation procedure based on back-propagation-through-time.

2.1 Differentiable synapse model

In spiking networks, the transmission of neural activity is mediated by synaptic current. Most models describe the synaptic current dynamics as a linear filter process, which instantly activates when the presynaptic membrane voltage v crosses a threshold: *e.g.*,

$$\tau \dot{s} = -s + \sum_k \delta(t - t_k). \tag{1}$$

where $\delta(\cdot)$ is the Dirac-delta function, and t_k denotes the time of threshold-crossing. Such threshold-triggered dynamics generates discrete, all-or-none responses of synaptic current, which is non-differentiable.

Here, we replace the threshold with a gate function $g(v)$: a non-negative ($g \geq 0$), unit integral ($\int g \, dv = 1$) function with narrow support[1], which we call the active zone. This allows the synaptic current to be activated in a gradual manner throughout the active zone. The corresponding synaptic current dynamics is

$$\tau \dot{s} = -s + g\dot{v}, \tag{2}$$

where $\dot{v}$ is the time derivative of pre-synaptic membrane voltage. The $\dot{v}$ term is required for the dimensional consistency between eq (1) and (2): The $g\dot{v}$ term has the same $[\text{time}]^{-1}$ dimension as the Dirac-delta impulses of eq (1), since the gate function has the dimension $[\text{voltage}]^{-1}$ and $\dot{v}$ has the dimension $[\text{voltage}][\text{time}]^{-1}$. Hence, the time integral of synaptic current, *i.e.* charge, is a dimensionless quantity. Consequently, a depolarization event beyond the active zone induces a constant amount of total charge regardless of the time scale of presynaptic depolarization, since

$$\int s \, dt = \int g\dot{v} \, dt = \int g \, dv = 1.$$

Therefore, eq (2) generalizes the threshold-triggered synapse model while preserving the fundamental property of spiking neurons: *i.e.* all supra-threshold depolarizations induce the same amount of synaptic responses regardless of the depolarization rate (Figure 1A,B). Depolarizations below the active zone induce no synaptic responses (Figure 1E), and depolarizations within the active zone induce graded responses (Figure 1C,D). This contrasts with the threshold-triggered synaptic dynamics, which causes abrupt, non-differentiable change of response at the threshold (Figure 1, dashed lines).

Note that the $g\dot{v}$ term reduces to the Dirac-delta impulses in the zero-width limit of the active zone, which reduces eq (2) back to the threshold-triggered synapse model eq (1).

[1] Support of a function $g : X \to \mathbb{R}$ is the subset of the domain X where $g(x)$ is non-zero.

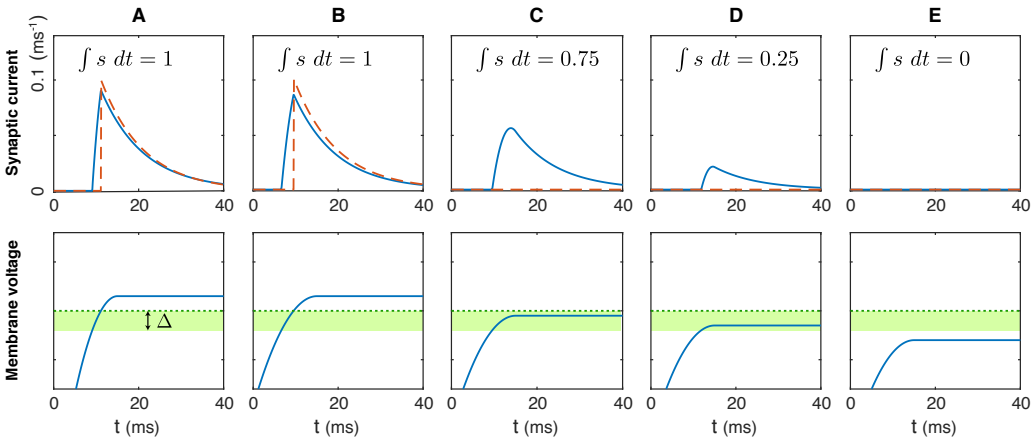

Figure 1: Differentiability of synaptic current dynamics: The synaptic current traces from eq (2) (solid lines, upper panels) are shown with the corresponding membrane voltage traces (lower panels). Here, the gate function is $g = 1/\Delta$ within the active zone of width Δ (shaded area, lower panels); $g = 0$ otherwise. (A,B) The pre-synaptic membrane voltage depolarizes beyond the active zone. Despite the different rates of depolarization, both events incur the same amount of charge in the synaptic activity: $\int s\ dt = 1$. (C,D,E) Graded synaptic activity due to insufficient depolarization levels that do not exceed the active zone. The threshold-triggered synaptic dynamics in eq (1) is also shown for comparison (red dashed lines, upper panels). $\tau = 10$ ms. The effect of voltage reset is ignored for the purpose of illustration.

The gate function, without the $\dot{v}$ term, was previous used as a differentiable model of synaptic connection [15]. In such a model, however, a spike event delivers varying amount of charge depending on the presynaptic depolarization rate: the slower the presynaptic depolarization, the greater the amount of charge delivered to the post-synaptic targets.

2.2 Network model

To complete the input-output dynamics of a spiking neuron, the synaptic current dynamics eq (2) must be coupled with the presynaptic neuron's internal state dynamics. For simplicity, we consider differentiable neural dynamics that depend only on the the membrane voltage and the input current:

$$\dot{v} = f(v, I). \tag{3}$$

The dynamics of an interconnected network of neurons can then be constructed by linking the dynamics of individual neurons and synapses eq (2,3) through the input-current vector:

$$\vec{I} = W\vec{s} + U\vec{i} + \vec{I}_o, \tag{4}$$

where W is the recurrent connectivity weight matrix, U is the input weight matrix, $\vec{i}(t)$ is the input signal for the network, and $\vec{I}_o$ is the tonic current. Note that this formulation describes general, fully connected networks. Specific network structures can be imposed by constraining the connectivity: *e.g.* triangular matrix structure W for multi-layer feedforward networks.

Lastly, we define the output of the network as the linear readout of the synaptic current:

$$\vec{o}(t) = O\vec{s}(t),$$

where O is the readout matrix.

The network parameters W, U, O, $\vec{I}_o$ will be optimized to minimize the total cost, $C \equiv \int l(t)\ dt$, where l is the cost function that evaluates the performance of network output for given task.

2.3 Gradient calculation

The above spiking neural network model can be optimized via gradient descent. In general, the exact gradient of a dynamical system can be calculated using either Pontryagin's minimum principle [16],

also known as backpropagation through time, or real-time recurrent learning, which yield identical results. We present the former approach here, which scales better with network size: $\mathcal{O}(N^2)$ instead of $\mathcal{O}(N^3)$.

Backpropagation through time for the spiking dynamics eq (2,3) utilizes the following backpropagating dynamics of adjunct state variables:

$$-\dot{p}_v = f_v p_v - g\dot{p}_s \tag{5}$$

$$-\tau\dot{p}_s = -p_s + \xi, \tag{6}$$

where $f_v \equiv \partial f/\partial v$, and ξ is called the *error current*. For the recurrently connected network eq (4), the error current vector has the following expression

$$\vec{\xi} = W^\intercal(\vec{f_I p_v}) + \vec{\partial}_s l, \tag{7}$$

which links the backpropagating dynamics eq (5,6) of individual neurons. Here, $f_I \equiv \partial f/\partial I$, $(f_I p_v)_j \equiv f_{I_j} p_{v_j}$, and $(\partial_s l)_j \equiv \partial l/\partial s_j$ (See Supplementary Materials).

Interestingly, the coupling term of the backpropagating dynamics, $g\dot{p}_s$, has the same form as the coupling term $g\dot{v}$ of the forward-propagating dynamics. Thus, the same gating mechanism that mediates the spiked-based communication of signals also controls the propagation of error in the same sparse, compressed manner.

Given the adjunct state vectors that satisfy eq (5,6,7), the gradient of the total cost with respect to the network parameters can be calculated as

$$\nabla_W C = \int (\vec{f_I p_v})\ \vec{s}^\intercal\ dt \tag{8}$$

$$\nabla_U C = \int (\vec{f_I p_v})\ \vec{i}^\intercal\ dt$$

$$\nabla_{I_o} C = \int (\vec{f_I p_v})\ dt$$

$$\nabla_O C = \int \vec{\partial}_o l\ \vec{s}^\intercal\ dt$$

where $(\partial_o l)_j \equiv \partial l/\partial o_j$. Note that the gradient calculation procedure involves multiplication between the presynaptic input source and the postsynaptic adjunct state p_v, which is driven by the $g\dot{p}_s$ term: *i.e.* the product of postsynaptic spike activity and temporal difference of error. This is analogous to reward-modulated spike-time dependent plasticity (STDP) [17].

3 Results

We demonstrate our method by training spiking networks on dynamic tasks that require information processing over time. Tasks are defined by the relationship between time-varying input-output signals, which are used as training examples. We draw mini-batches of ≈ 50 training examples from the signal distribution, calculate the gradient of the average total cost, and use stochastic gradient descent [18] for optimization.

Here, we use a cost function l that penalizes the readout error and the overall synaptic activity:

$$l = \frac{\|\vec{o} - \vec{o}_d\|^2 + \lambda\|\vec{s}\|^2}{2},$$

where $\vec{o}_d(t)$ is the desired output, and λ is a regularization parameter.

3.1 Predictive Coding Task

We first consider predictive coding tasks [14, 19], which optimize spike-based representations to accurately reproduce the input-ouput behavior of a linear dynamical system with full-rank input and output matrices. Analytical solutions for this class of problems can be obtained in the form of non-leaky integrate and fire (NIF) neural networks, although insignificant amount of leak current is often added. The solutions also require the networks to be equipped with a set of instantaneously fast synapses in addition to regular synapses with finite time constant [19].

The membrane voltage dynamics of a NIF neuron is given by

$$f(v, I) = I.$$

To ensure that the membrane voltage stay within a finite range, we imposed two thresholds at $v_{\theta+} = 1$ and $v_{\theta-} = -1$, and the reset voltage at $v_{\text{reset}} = 0$[2]. For simplicity, we allow the $v_{\theta-}$ threshold to trigger negative synaptic responses, which can be turned off if desired.

We also introduce the additional fast synaptic current $\vec{s}_f$ proposed in [14, 19, 13], which modifies the input current vector to be $\vec{I} = W\vec{s} + W_f\vec{s}_f + U\vec{i} + \vec{I}_o$, where W_f is the recurrent weight matrix associated with fast synapses. However, assigning zero time constant to fast synapses often causes unstable dynamics, because it could lead to one spike immediately triggering spikes in other neurons. Here, we assign finite time constants for both types of synapses: $\tau_f = 1$ ms for fast synapses, and $\tau = 10$ ms for regular synapses.

Despite its simplicity, the predictive coding framework reproduces important features of biological neural networks, such as the balance of excitatory and inhibitory inputs and efficient coding [14]. Also, its analytical solutions provide a benchmark for assessing results from optimization.

Auto-encoder task In the *auto-encoder* task, the desired output signal is a low-pass filtered version of the input signal:

$$\tau\dot{\vec{o}}_d = -\vec{o}_d + \vec{i},$$

where τ is the synaptic time constant [14, 19]. The goal is to accurately represent the analog signals using least number of spikes. We used a network of 30 NIF neurons, and 2 input and output signals. Randomly generated sum-of-sinusoid signals with period 1200 ms were used as the input. $\lambda = 0.1/N$ ms^2. $\Delta = 0.1$ was used for training, then set to zero for post-training simulations.

The output of the trained network accurately tracks the desired output (Figure 2A). Analysis of the simulation reveals that the network operates in a tightly balanced regime: The fast recurrent synaptic input from other neurons, $W_f\vec{s}_f(t)$, provides equal and opposite current that mostly cancels the input current from the external signal, $U\vec{i}(t)$, such that the neuron generates a greatly reduced number of spike outputs (Figure 2B,C,D). The network structure also shows close agreement to the prediction. The optimal input weight matrix is equal to the transpose of the readout matrix (up to a scale factor), $U \propto O^\intercal$, and the optimal fast recurrent weight is approximately the product of the input and readout weights, $W_f \approx -UO$, which are in close agreement with [14, 19, 20]. The regular recurrent connection is not needed for this task and hence W was set to zero. Such network structures are shown to maintain tight input balance and remove redundant spikes to encode the signals in most efficient manner: The representation error scales as $1/K$, where K is the number of involved spikes, compared to the $1/\sqrt{K}$ error of encoding with independent Poisson spikes.

General predictive coding task More generally, predictive coding tasks involve linear dynamic relationships between the desired input-output signals of the following form:

$$\tau\dot{\vec{o}}_d = -\vec{o}_d + A\vec{o}_d + \vec{i},$$

where A is a constant matrix. Here, we trained a spiking network of 30 NIF neurons with 2 input signals of sums-of-sinusoid and $A = [-0.7, 0.36; -2.3, -0.1]$, which strongly modulates the desired output signal dynamics.

Similar to the result shown in Figure 2, the trained network exhibits tightly balanced input current with the network output accurately tracking the desired output. The optimal *regular* recurrent weight is approximately $W \approx UAO$ (Figure 3), which is also in close agreement with the prediction [14, 19, 20]. The other network structures are similar to the case of auto-encoding task.

These results show that our algorithm accurately optimizes the millisecond time-scale interaction between neurons to find an efficient spike-time-based encoding scheme. Moreover, it also shows that efficient coding can be robustly achieved without introducing instantaneously fast synapses, which were previously considered to be necessary.

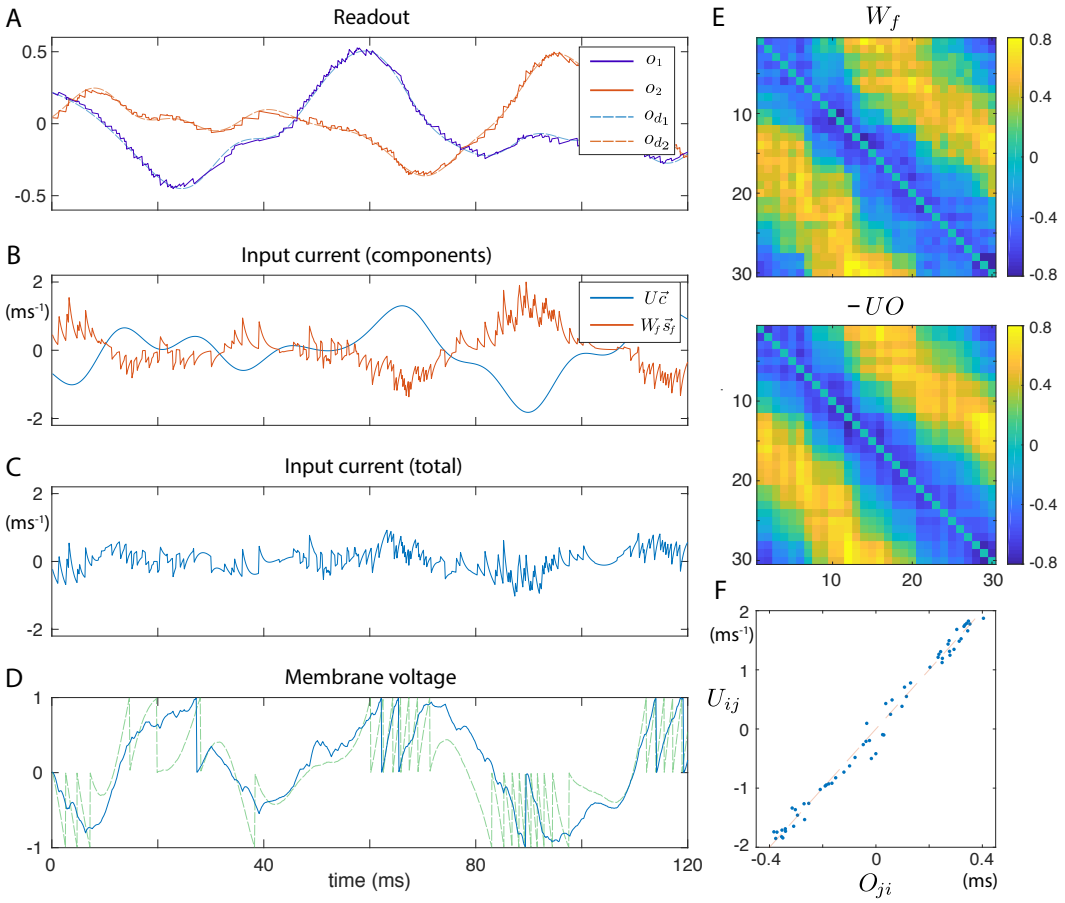

Figure 2: Balanced dynamics of a spiking network trained for auto-encoding task. (A) Readout signals: actual (solid), and desired (dashed). (B) Input current components into a single neuron: external input current ($U\vec{i}(t)$, blue), and fast reccurent synaptic current ($W_f\vec{s}_f(t)$, red). (C) Total input current into a single neuron. (D) Single neuron membrane voltage traces: An actual voltage trace driven by both external input and fast reccurent synaptic current (solid, 6 spikes), and a virtual trace driven by external input only (dashed, 29 spikes). (E) Fast recurrent weight: trained (W_f, up) and predicted ($-UO$, down). Diagonal elements are set to zero to avoid self-excitation/inhibition. (F) Readout weight O vs input weight U.

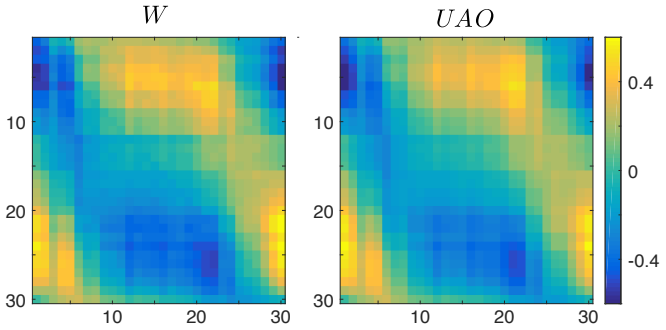

Figure 3: Regular recurrent weight for the predictive coding task: trained (W, left) and predicted (UAO, right).

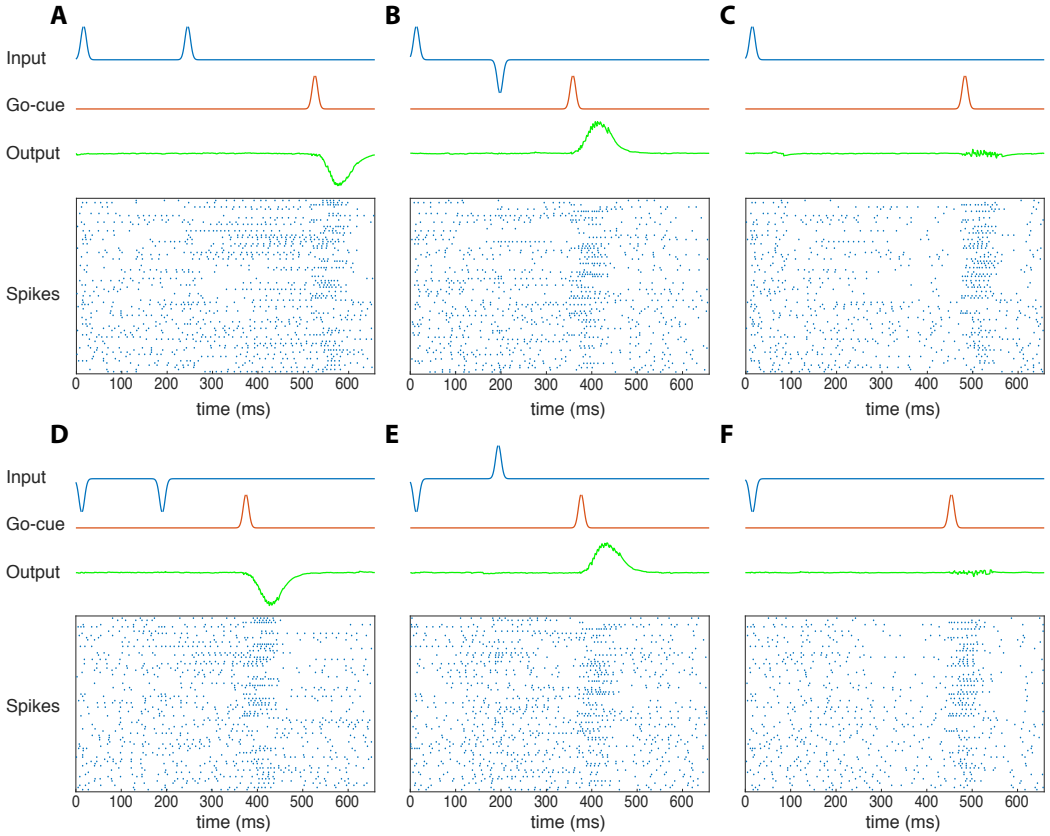

Figure 4: Temporal XOR task: Each panel shows the single-trial input, go-cue, output traces, and spike raster of an optimized QIF neural network. The y-axis of the raster plot is the neuron ID. Note the similarity of the initial portion of spike patterns for trials with the same first input pulses (A,B,C vs D,E,F). In contrast, the spike patterns after the go-cue signal are similar for trials with the same desired output pulses (A,D: negative output), (B,E: positive output), and (C,F: null output).

3.2 Temporal XOR task

A major challenge for spike-based computation is in bridging the wide difference between the time-scales of behavior and spikes: How do millisecond spikes perform behaviorally relevant computations on the order of seconds?

Here, we consider a delayed-memory XOR task, which requires storing the input pulses over long time delays, then performing the exclusive-or operation on the stored history. Specifically, the network receives binary pulse signals, $+$ or $-$, through an input channel and a go-cue through another channel. If the network receives two input pulses since the last go-cue signal, it should generate the XOR output pulse on the next go-cue: *i.e.* a positive output pulse if the input pulses are of opposite signs ($+-$ or $-+$), and a negative output pulse if the input pulses are of equal signs ($++$ or $--$). Additionally, it should generate a null output if only one input pulse is received since the last go-cue signal. Variable time delays are introduced between the input pulses and the go-cues.

A simpler version of the task was proposed in [13], whose solution involved first training a small analog, rate-based neural network and transferring the learned network dynamics to a larger (≈ 3000) network of spiking neurons, using the method of predictive coding scheme [14]. It also required a dendritic nonlinearity function to match the transfer function of rate neurons.

[2]The reset process may seem non-differentiable, but in fact it does not influence the gradient calculation.

We trained a network of 80 QIF (quadratic integrate and fire) neurons[3], whose dynamics is

$$f(v, I) = (1 + \cos(2\pi v))/\tau_v + (1 - \cos(2\pi v))I$$

(in canonical, theta model form) with the threshold and the reset voltage at $v_\theta = 1$, $v_{\text{reset}} = 0$. Time constants of $\tau_v = 25$, $\tau_f = 5$, and $\tau = 20$ ms were used, whereas the time-scale of the task was ≈ 500 ms, much longer than the time constants. The intrinsic nonlinearity of the QIF spiking dynamics proves to be sufficient to solve this task without requiring extra dendritic nonlinearity. The trained network successfully solves the XOR task (Figure 4): The spike patterns exhibit time-varying, but sustained activities that maintain the input history, generate the correct outputs when triggered by the go-cue signal, then return to the background activity. More analysis is needed to understand the exact underlying computational mechanism.

This result shows that out algorithm can indeed optimize spiking networks to perform nonlinear computations over extended time.

4 Discussion

We have presented a novel, differentiable formulation of spiking neural networks and derived the gradient calculation for supervised learning. Unlike previous learning methods, our method optimizes the spiking network dynamics for general supervised tasks on the time scale of individual spikes as well as behavioral time scales.

Exact gradient-based learning methods inevitably involve discrepancies from biological learning processes. Nonetheless, such methods provide solid theoretical ground for understanding the principles of biological learning rules. For example, our result shows that the gradient update occurs in a sparsely compressed manner near the spike times, bearing close resemblance to reward-modulated STDP. Moreover, further analysis may reveal that certain aspects of the gradient calculation can be approximated in a biologically plausible manner without significantly compromising the efficiency of optimization. For example, it was recently shown that the biologically implausible aspects of backpropagation method can be resolved through feedback alignment for rate-based multilayer feedforward networks [21]. Such approximations could also apply to spiking neural networks.

Here, we coupled the differentiable synaptic current model eq (2) with differentiable single-state spiking neuron models eq (3). However, the synapse model can be coupled with any neuron models, including realistic multi-state neuron models with detailed action potential dynamics[4], including the Hodgkin-Huxley model, the Morris-Lecar model, and the FitzHugh-Nagumo model; and even models with internal adaptation variables. It can also be coupled with neuron models having non-differentiable reset dynamics, such as the leaky integrate and fire model, the exponential integrate and fire model, and the Izhikevich model, although gradient calculation on these models would require additional procedures. This will be examined in the future work.

Acknowledgments

We thank Peter Dayan for helpful discussions.

References

[1] Valerio Mante, David Sussillo, Krishna V Shenoy, and William T Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503(7474):78–84, 2013.

[2] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. *Proceedings of the National Academy of Sciences*, 111(23):8619–8624, 2014.

[3] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521(7553):436–444, 2015.

[4] Sander M Bohte, Joost N Kok, and Han La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1):17–37, 2002.

[3] NIF networks fail to learn the temporal XOR task: the memory requirement for past input history drives the training toward strong recurrent connections and runaway excitation.

[4] Simple modification of the gate function would be required to prevent activation during the falling phase of action potential.

[5] Raoul-Martin Memmesheimer, Ran Rubin, Bence P Ölveczky, and Haim Sompolinsky. Learning precisely timed spikes. *Neuron*, 82(4):925–938, 2014.

[6] Robert Gütig and Haim Sompolinsky. The tempotron: a neuron that learns spike timing–based decisions. *Nature neuroscience*, 9(3):420–428, 2006.

[7] Jean-Pascal Pfister, Taro Toyoizumi, David Barber, and Wulfram Gerstner. Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning. *Neural computation*, 18(6):1318–1348, 2006.

[8] Răzvan V Florian. Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity. *Neural Computation*, 19(6):1468–1502, 2007.

[9] Eugene M Izhikevich. Solving the distal reward problem through linkage of stdp and dopamine signaling. *Cerebral cortex*, 17(10):2443–2452, 2007.

[10] Robert Legenstein, Dejan Pecevski, and Wolfgang Maass. A learning theory for reward-modulated spike-timing-dependent plasticity with application to biofeedback. *PLoS Comput Biol*, 4(10):e1000180, 2008.

[11] Filip Ponulak and Andrzej Kasiński. Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting. *Neural Computation*, 22(2):467–510, 2010.

[12] Eric Hunsberger and Chris Eliasmith. Spiking deep networks with lif neurons. *arXiv preprint arXiv:1510.08829*, 2015.

[13] LF Abbott, Brian DePasquale, and Raoul-Martin Memmesheimer. Building functional networks of spiking model neurons. *Nature neuroscience*, 19(3):350–355, 2016.

[14] Sophie Denève and Christian K Machens. Efficient codes and balanced networks. *Nature neuroscience*, 19(3):375–382, 2016.

[15] Guillaume Lajoie, Kevin K Lin, and Eric Shea-Brown. Chaos and reliability in balanced spiking networks with temporal drive. *Physical Review E*, 87(5):052901, 2013.

[16] Lev Semenovich Pontryagin, EF Mishchenko, VG Boltyanskii, and RV Gamkrelidze. The mathematical theory of optimal processes. 1962.

[17] Nicolas Frémaux and Wulfram Gerstner. Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules. *Frontiers in neural circuits*, 9, 2015.

[18] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Martin Boerlin, Christian K Machens, and Sophie Denève. Predictive coding of dynamical variables in balanced spiking networks. *PLoS Comput Biol*, 9(11):e1003258, 2013.

[20] Wieland Brendel, Ralph Bourdoukan, Pietro Vertechi, Christian K Machens, and Sophie Denéve. Learning to represent signals spike by spike. *arXiv preprint arXiv:1703.03777*, 2017.

[21] Timothy P Lillicrap, Daniel Cownden, Douglas B Tweed, and Colin J Akerman. Random synaptic feedback weights support error backpropagation for deep learning. *Nature Communications*, 7, 2016.

Supplementary Materials: Gradient calculation for the spiking neural network

Pontryagin's minimum principle According to [16], the Hamiltonian for the spiking network dynamics eq (2,3,4) is

$$\begin{aligned}\mathcal{H} &= \sum_i \bar{p}_{v_i}\dot{v}_i + \bar{p}_{s_i}\dot{s}_i + l(\vec{s}) \\ &= \sum_i (\bar{p}_{v_i} + g(v_i)\bar{p}_{s_i}/\tau) f(v_i, I_i) - (\bar{p}_{s_i}/\tau)s_i + l(\vec{s}),\end{aligned}$$

where $\bar{p}_{v_i}$ and $\bar{p}_{s_i}$ are the adjunct state variables for the membrane voltage v_i and the synaptic current s_i of neuron i, respectively, and $l(\vec{s})$ is the cost function. The back-propagating dynamics of the adjunct state variables are:

$$\begin{aligned}-\dot{\bar{p}}_{v_i} &= \frac{\partial \mathcal{H}}{\partial v_i} = (\bar{p}_{v_i} + g_i\bar{p}_{s_i}/\tau) f_{v_i} + f_i g_i' \bar{p}_{s_i}/\tau \\ -\dot{\bar{p}}_{s_i} &= \frac{\partial \mathcal{H}}{\partial s_i} = \sum_j (\bar{p}_{v_j} + g_j\bar{p}_{s_j}/\tau)\cdot f_{I_j} W_{ji} - \bar{p}_{s_i}/\tau + l_{s_i}\end{aligned}$$

where $f_v \equiv \partial f/\partial v$, $f_I \equiv \partial f/\partial I$, $g' \equiv dg/dv$, and $l_{s_i} \equiv \partial l/\partial s_i$.

This formulation can be simplified by change of variables, $p_v \equiv \bar{p}_v + g\bar{p}_s/\tau$, $p_s \equiv \bar{p}_s/\tau$, which yields

$$\begin{aligned}\mathcal{H} &= \vec{p}_v \cdot \vec{f} - \vec{p}_s \cdot \vec{s} + l \\ -\dot{p}_{v_i} &= f_{v_i} p_{v_i} - g_i \dot{p}_{s_i} \\ -\tau\dot{p}_{s_i} &= -p_{s_i} + l_{s_i} + \sum_j W_{ji} f_{I_j} p_{v_j},\end{aligned}$$

where we used $\dot{p}_{v_i} = \dot{\bar{p}}_{v_i} + f_i g_i' \bar{p}_{s_i}/\tau + g_i \dot{\bar{p}}_{s_i}/\tau$.

The gradient of the total cost can be obtained by integrating the partial derivative of the Hamiltonian with respect to the parameter (*e.g.* $\partial\mathcal{H}/\partial W_{ij}$, $\partial\mathcal{H}/\partial U_{ij}$, $\partial\mathcal{H}/\partial I_{o_i}$, $\partial\mathcal{H}/\partial O_{ij}$).