Journal of Machine Learning Research 22 (2021) 1-21 Submitted 9/20; Revised 5/21; Published 11/21

Bifurcation Spiking Neural Network

Shao-Qun Zhang ZHANGSQ@LAMDA.NJU.EDU.CN
Zhao-Yu Zhang ZHANGZHAOYU@LAMDA.NJU.EDU.CN
Zhi-Hua Zhou ZHOUZH@LAMDA.NJU.EDU.CN
National Key Laboratory for Novel Software Technology
Nanjing University
Nanjing 210023, China

Editor: George Konidaris

Abstract

Spiking neural networks (SNNs) have attracted much attention due to their great potential for modeling time-dependent signals. The performance of SNNs depends not only on picking an apposite architecture and searching optimal connection weights as well as conventional deep neural networks, but also on the careful tuning of many hyper-parameters within fundamental spiking neural models. However, so far, there has been less systematic work on analyzing SNNs' dynamical characteristics, especially ones relative to these *internal* hyper-parameters, which leads to whether SNNs are adequate for modeling actual data relies on fortune. In this work, we provide a theoretical framework for investigating spiking neural models from the perspective of dynamical systems. As a result, we point out that the LIF model with control rate hyper-parameters is a *bifurcation* dynamical system. This point explains why the performance of SNNs is so sensitive to the setting of control rate hyper-parameters, leading to a recommendation that diverse and adaptive eigenvalues are beneficial to improve the performance of SNNs. Inspired by this insight, we develop the *Bifurcation Spiking Neural Network* (BSNN) with supervised implementation, and theoretically show that BSNN is an adaptive dynamical system. Experiments validate the effectiveness of BSNN on several benchmark data sets, showing that BSNN achieves superior performance to existing SNNs and is robust to the setting of control rates.

Keywords: Spiking Neural Network, Leaky Integrate-and-Fire model, Control Rates, Eigenvalues, Bifurcation Dynamical System

1. Introduction

Spiking neural networks (SNNs) take into account the time of spike firing rather than simply relying on the accumulated signal strength in conventional neural networks, and thus offering the possibility for modeling time-dependent data (Shimokawa et al., 1999; VanRullen et al., 2005). The fundamental spiking neural model is usually formulated as a first-order parabolic equation with many biologically realistic (*i.e.*, internal) hyper-parameters, *e.g.*, the membrane resistance and control rate hyper-parameters in Leaky Integrate-and-Fire (LIF) model (Burkitt, 2006a,b) and the membrane capacitance hyper-parameter in Hodgkin-Huxley (HH) model (Gerstner and Kistler, 2002). Thus, the performance of SNNs depends not only on determining the neural network architecture and training connection weights as well as conventional deep neural networks but also on the careful tuning of these *internal* hyper-parameters. Existing SNNs often follow the bio-plausible knowledge in neuroscience to set these internal hyper-parameters and then make minor a tuning of some hyper-parameters (Hohn and Burkitt, 2001; Carlson et al., 2014). Usually, the traditional manners look clumsy and have unrobust performance with large computation due to the lack of systematic analysis on designating which hyper-parameters are sensitive and guiding how to tune them.

This work investigates the dynamical properties of the LIF-modeling SNNs, especially the influence of hyper-parameters on the model dynamics. As a result, we declare that the LIF model is a bifurcation dynamical system, which means that its topology depends sensitively on the control rate hyper-parameters. This result sheds three significant insights: (1) The performance of SNNs is sensitive to the setting of control rates, which is consistent with the facts. (2) It is necessary to enable diverse and learnable control rates, corresponding to the eigenvalues of bifurcation dynamical systems, for achieving adaptive systems. This claim argues the conventional manners that the control rates are neatly preset as a negative fixed value. (3) The role of control rates cannot be replaced by learnable connection parameters and other hyper-parameters.

However, training control rates is a very tricky challenge. Since control rates and connection weights are entangled during the training process, the approaches (Hunter et al., 2012; Lorenzo et al., 2017) of turning hyper-parameters in conventional neural networks cannot be directly used to solve this issue. An alternative way is to sample the control rates from a certain pre-defined distribution and find the optimal ones by alternating optimization. Nevertheless, this method usually succeeds on an apposite distribution and larger computation and storage.

To tackle the challenges above and improve the performance of SNNs, we propose the *Bifurcation Spiking Neural Network* (BSNN). By exploiting the bifurcation theory, we convert the issue of learning a group of adaptive control rates into a new problem of learning a collection of apposite eigenvalues. So BSNN overcomes the obstacle that controls rates interact with connection weights, leading to a robust control rate setting and achieves a laudable performance with considerably less computation and storage than the alternating optimization approaches. The experiments conducted on four benchmark data sets demonstrate the effectiveness of BSNN, showing that its performance surpasses existing SNNs and is robust to the setting of control rates.

Our main contributions are summarized as follows:

- We provide a theoretical framework for studying the dynamical properties of spiking neural models, *e.g.*, we show the LIF model is a bifurcation dynamical system in Section 4.
- We point out the fact that the control rate hyper-parameter, rather than other ones, is sensitive to the performance of SNNs with LIF neurons.
- We present the BSNN with supervised implementation, and then theoretically show that the dynamical system led by BSNN has adaptive eigenvalues, leaving a robust setting of the control rate hyper-parameters.
- We perform numerical studies on several data sets to demonstrate the conclusions above and advantages of our proposed models.

The rest of this paper is organized as follows. Section 2 reviews the related works. Section 3 introduces some preliminary knowledge about spiking neural models. Section 4 establishes a theoretical analysis for SNNs based on dynamical system theory and provides the alternating optimization approaches for achieving adaptive systems. Section 5 formally presents the BSNN with theoretical analysis and concrete implementation. The experiments are conducted in Section 6. Finally, Section 7 concludes our work with prospects.

2. Related Works

The last two decades have witnessed the increasing prominence of SNNs in machine learning and artificial intelligence research, leading to a proliferation of efficient software packages for their training and deployment. Existing methods for training SNNs can be roughly divided into three categories. The first category leverages the training procedure by exploring a simple continuous-valued artificial neural network and converting this deep network to accurate spiking equivalents (O'Connor et al., 2013; Rueckauer et al., 2017). The second category configures an SNN on discontinuous spike activities. For example, SpikeProp and its varients transfer the information in the timing of a single spike (Bohte et al., 2002; McKennoch et al., 2006). However, SpikeProps are limited to single-spike learning, which usually cause a large number of neurons to be in a shutdown state. Recently, Jin et al. (2018) account for the temporal contribution of the given pre-synaptic spike train to the firings of the post-synaptic neuron. The third category attempts to train SNNs in a temporal manner. For example, Huh and Sejnowski (2018) present a temporal gradient descent method with a differentiable formulation of spiking dynamics. Shrestha and Orchard (2018) incorporate the temporal dependency between spikes, and thus the back-propagated error at a given time step can be written as a integration of earlier spike inputs.

As mentioned above, the performance of SNNs depends not only on determining the neural network architecture and training connection weights as well as conventional deep neural networks but also on the careful tuning of these internal hyper-parameters. Existing SNNs, such as Hohn and Burkitt (2001), Jin et al. (2018), and Zhang and Li (2019), often follow the bio-plausible knowledge in neuroscience to set these internal hyper-parameters and then make a fine-tuning of all hyper-parameters. Carlson et al. (2014) employed evolutionary algorithms to optimize the hyper-parameters within spiking neural models. Kulkarni and Rajendran (2018) described several hyper-parameter tuning experiments achieved on the MNIST data set. However, there still lacks apposite theoretical analysis for guiding whether all hyper-parameters are important, which hyper-parameter is sensitive, and how to tune them. Our work, thanks to dynamical system theory, provides a theoretical framework for investigating spiking neural models. We point out the dynamical properties of SNNs with different neuron models and give positive suggestions on the setting of sensitive hyper-parameters.

Zero-order alternating optimization has been widely discussed in neural-network-related researches. These approaches can roughly be divided into two categories: (1) Optimizing the architectures of neural networks. For example, Lorenzo et al. (2017) use particle swarm optimization to select the architectures of DNNs. (2) Optimizing internal hyper-parameters. Bergstra and Bengio (2012) provide a classical way that randomly samples hyper-parameters from a prior distribution and finds the best one. However, due to the limitation of observation information, the zero-order alternating optimization approaches intrinsically require large computation and storage.

3. Spiking Neural Model

In this work, we focus more on the LIF model, not only due to the ease with which the LIF model can be analyzed and simulated but also because this model is the most widely used in the intersection of SNN and Artificial Intelligence. Here, we review a general form of the LIF model as follows:

$$\tau_m \frac{\mathrm{d}u}{\mathrm{d}t} = -u + R\, f(\boldsymbol{I}(\boldsymbol{t})), \tag{1}$$

where $\boldsymbol{I}(t) = (I_1(t), \ldots, I_M(t))^\top$ denotes the M-dimensional input signals, $u(t)$ indicates the membrane potential of the concerned neuron at time t, τ_m and R are positive-valued hyper-parameters with respect to membrane time and membrane resistance, respectively, and f represents the aggregation function, usually expressed in a linear formation, $f(\boldsymbol{I(t)}) = \sum_{j=1}^{M} \mathbf{W}_j I_j(t)$, in which $\mathbf{W}_j$ denotes the learnable connection weight corresponding to the j-th input channel. Based on the *spike response model* scheme (Gerstner, 1995), the LIF equation has a general solution with the boundary condition $u_{rest} = 0$ as follows:

$$u(t) = \sum_{j=1}^{M} \mathbf{W}_j \left[\int_{t'}^{t} \exp\left(\frac{t' - s}{\tau_m} \right) I_j(s) \,\mathrm{d}s \right], \tag{2}$$

where t' denotes the last firing time $t' = \max\{s \mid u(s) = u_{firing}, s < t\}$. An output stimulus $S(t)$ is generated whenever the membrane potential $u(t)$ reaches a certain threshold u_{firing} (firing threshold). We formulate this procedure using a spike excitation function

$$f_e : u \to S, \quad \text{where} \quad S(t) \triangleq \left\lfloor \frac{u(t)}{u_{firing}} \right\rfloor .$$

After firing, the membrane potential is instantaneously reset to a lower value u_{rest} (rest voltage). Some researchers take the delayed-firing behaviors in neuroscience, ***i.e.***, add an absolute refractory period Δ_{abs} (Hunsberger and Eliasmith, 2015) or a refractory kernel $\eta_{ref}(\tau_m)$ (Dumont et al., 2017) to the LIF model. Thus, Eq. (2) becomes

$$u(t) = \sum_{j=1}^{M} \mathbf{W}_j \left[\int_{t'}^{t} \exp\left(\frac{t' - s}{\tau_m} \right) I_j(s) \,\mathrm{d}s \right] + \eta_{ref}(\tau_m) * S(t), \tag{3}$$

where $*$ denotes the convolution operation and $\eta_{ref}(\tau_m) * S(t)$ is the refractory response of a neuron.

Notice that the LIF model with the initial condition $u(0) = u_{rest}$ or $u(t') = u_{rest}$ leads to an ordinary differential equation dynamical system. The eigenvalue ρ of this dynamical system is equal to the quotient of -1 and τ_m by solving the following algebraic equations

$$\begin{cases} \dfrac{\mathrm{d}u}{\mathrm{d}t} = -\dfrac{1}{\tau_m} u, \\ u(0) = u_{rest}. \end{cases}$$

Subsection 3.2 details the calculation of the eigenvalues of ordinary differential equations.

3.1 Neural Encoding

Before fed up to SNNs, the real input data (*e.g.*, image or video) should be pre-converted into a spiking version. The conversion procedure is called *neural encoding*. There are two main categories of neural encoding approaches, that is, *temporal encoding* and *rate-based encoding*. In temporal encoding (Yu et al., 2014), the input data is encoded by the distance between time instances that fire spikes. In rate-based encoding (Lobo et al., 2020), the input data is encoded by the number of fired spikes within temporal windows. The representative approaches are usually encoded by a Poisson distribution and recorded by a Dynamic Vision Sensor (Quiroga et al., 2005; Anumula et al., 2018), illustrated in Figure 1. Thus, the rate-based encoding becomes the simplest and most popular encoding scheme in SNNs. Throughout this paper, we adopt rate-based encoding as the default.

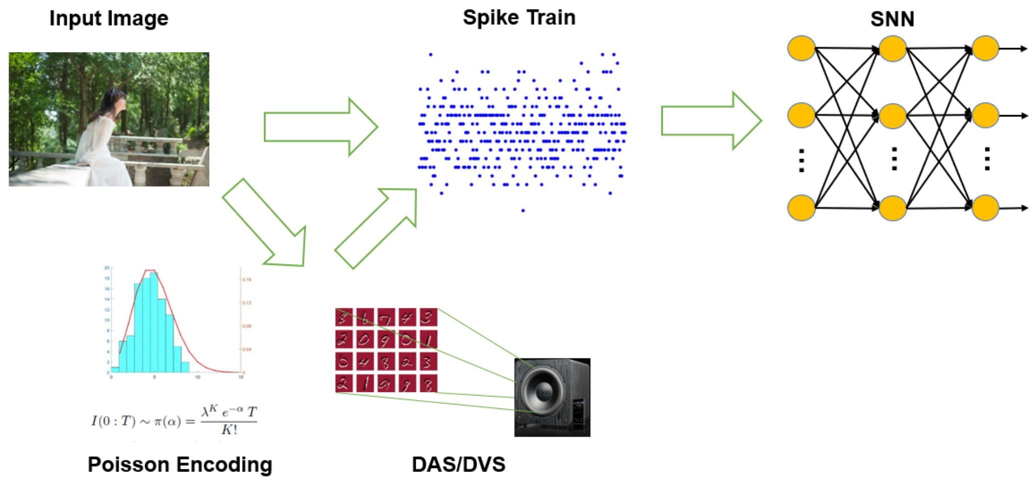

Input Image
Spike Train
SNN
$I(0:T) \sim \pi(\alpha) = \frac{\lambda^K e^{-\alpha} T}{K!}$
Poisson Encoding
DAS/DVS

Figure 1: Illustrations of rate-based neural encoding in spiking versions.

3.2 Eigenvalues and Abstract Equations

For a system of first-order linear differential equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \begin{pmatrix} \alpha_{11} & \dots & \alpha_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \dots & \alpha_{nn} \end{pmatrix} \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} + \begin{pmatrix} f_1(t) \\ \vdots \\ f_n(t) \end{pmatrix},$$

we have its algebraic formulation

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} = \mathbf{A} \cdot \begin{pmatrix} u_1 \\ \vdots \\ u_n \end{pmatrix} \quad \text{with} \quad \mathbf{A} = \begin{pmatrix} \alpha_{11} & \dots & \alpha_{1n} \\ \vdots & \ddots & \vdots \\ \alpha_{n1} & \dots & \alpha_{nn} \end{pmatrix}.$$

These algebraic equations are only related to the observation variables $u_1, \dots, u_n$. So the eigenvalues (spectrum values) of the matrix (operator) $\mathbf{A}$ lead to the evolution rules of the dynamical system.

4. Dynamical Properties of Spiking Neural Models

From Section 3, there are two internal hyper-parameters, *i.e.*, the membrane time τ_m and membrane resistance R in the LIF spiking neural model. In this section, we will investigate the dynamical properties of the LIF spiking neural models concerning the hyper-parameters above. For convenience, we define a new hyper-parameter $\gamma = -1/\tau_m$, called *control rate*.

4.1 Bifurcation Dynamics of LIF-modeling SNNs

Now, we present the first main theorem as follows:

Theorem 1 *Given the initial condition $u(0) = u_{rest}$ or $u(t') = u_{rest}$, the dynamical system led by one layer of LIF neurons is a bifurcation dynamical system, and control rate γ is the corresponding bifurcation hyper-parameter.*

According to the bifurcation theory in dynamical systems, a bifurcation occurs when a small smooth change of the parameter values (often the bifurcation hyper-parameters pass through a critical point) causes a sudden topological change in its behavior (Onuki, 2002; Kuznetsov, 2013). Theorem 1 shows that the bifurcation dynamical system led by one layer of LIF neurons is structurally unstable, and its performance is sensitive to the setting of the control rates.

Proof We start by formulating the total energy of the LIF model. From the perspective of dynamical systems, the LIF model with initial condition $u(0) = u_{rest}$ or $u(t') = u_{rest}$ guides a continuous dynamical system. Multiplying both sides of Eq. (1) by the membrane potential u, as commonly used by Hirsch et al. (2012), we can obtain the energy of a one-LIF-neuron dynamical system

$$\mathcal{H}(u,t) = u^2 + \frac{2R}{\tau_m} F(u,t) = u^2 - 2\gamma R F(u,t), \tag{4}$$

where u^2 indicates the kinetic energy and F is the primitive function of $f \cdot u$ that represents the non-potential forces. So $\mathcal{H}(u,t)$ is a Lyapunov-like function. Direct calculations show that:

$$\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t} = 2u\frac{\mathrm{d}u}{\mathrm{d}t} + \frac{2R}{\tau_m} f u = 2u\left(\frac{\mathrm{d}u}{\mathrm{d}t} + \frac{2R}{\tau_m} f\right) = -\frac{2}{\tau_m}u^2 = 2\gamma u^2. \tag{5}$$

Based on Eqs. (4) and (5), both the energy function $\mathcal{H}(u,\gamma,t)$ and its time derivative are dominated by the control rate. If the control rate is set as a negative constant as usual, we have $\mathrm{d}\mathcal{H}/\,\mathrm{d}t < 0$. Thus, the dynamical system led by one LIF neuron is energy-dissipating. This means that information will be continuously lost during the learning process of one LIF neuron and the concerned spiking neuron appears to hinder spike excitation. When $\gamma = 0$, one LIF neuron becomes a conservative system where the conversation from received signals to biological spikes is without damage, *i.e.*, the Integrate-and-Fire (IF) model. When $\gamma > 0$, the system energy would be increasing since $\mathrm{d}\mathcal{H}/\,\mathrm{d}t > 0$, and thus, the concerned neuron is encouraged to firing spikes.

Next, we extend the aforementioned results relative to one LIF neuron to a fully-connected feed-forward SNN. Similarly, by adding apposite initial conditions to the concerned spiking equations, the total energy of a layer of N LIF neurons can be given by

$$\mathcal{H}_N(\boldsymbol{u},t) = |\boldsymbol{u}|^2 + \frac{2R}{\tau_m} F(\boldsymbol{u},t), \tag{6}$$

where $\boldsymbol{u}$ is a N-dimensional vector that the i-th component represents the membrane potential of the i-th spiking neuron, and $F(\boldsymbol{u},t)$ is the primitive function of $\langle \boldsymbol{f}, \boldsymbol{u}\rangle$. The time derivative of the total energy function can be calculated by

$$\frac{\mathrm{d}\mathcal{H}_N}{\mathrm{d}t} = 2\gamma|\boldsymbol{u}|^2. \tag{7}$$

Similar to Eq. (5), the control rate is proportional to the eigenvalue ρ of LIF equation, *i.e.*, $\gamma = \rho$. So the time derivative of energy function, described by Eq. (5), becomes $\mathrm{d}\mathcal{H}_N/\,\mathrm{d}t = 2\rho|\boldsymbol{u}|^2$.

A bifurcation occurs once the control rate γ, as well as the eigenvalue ρ of this dynamical system, crosses the critical value 0; $\rho > 0$ leads to an unstable manifold, while $\rho < 0$ leads to a stable one. Therefore, $\mathcal{H}_N(\boldsymbol{u},t)$, described by Eq. (6), coincides with a bifurcation dynamical system concerning control rate γ. This completes the proof. ■

Based on the results above, the control rate γ relative to the membrane time hyper-parameter τ_m plays a crucial role on SNNs, directly affecting the topology of the dynamical system led by the LIF-based SNN. As conventionally preset γ to a negative value (since τ_m is often positive) in the whole network, SNNs comprise the stacking of dissipating systems. Thus, the energy will be continuously lost layer by layer during the learning process of SNNs. A better way is to adaptively learn the control rate so that SNNs can determine the desired system mode (*i.e.*, dissipating, conservative, diffusion systems, or a mixture of the three) according to the actual environment. On the other hand, existing SNNs usually employ unified control rates, that is, $\gamma_1 = \cdots = \gamma_N$. According to Theorem 1 and Eq. (7), this manner makes the topology structure of whole SNNs tied to only one pre-given hyper-parameter γ, thus greatly weakening the representation ability of SNNs. To revise this manner, we force the control rates in one layer to be diverse, *i.e.*, each neuron has an exclusive control rate parameter. In summary, the traditional manners that employ unified control rates and preset all control rates to one fixed and negative constant impede the flexibility of SNNs. To achieve adaptive systems, the control rates in SNNs should be diverse and learnable.

Notice that the change of the membrane resistance hyper-parameter R and connection weights $\mathbf{W}$ only affect the system energy from Eqs. (4) and (6), but has nothing to do with their intrinsic dynamical properties according to Eqs. (5) and (7). Therefore, the role of control rates cannot be replaced by learnable connection weights and other hyper-parameters.

4.2 Approaches for Parameterizing Control Rates

An intuitive idea for achieving adaptive SNN systems is to parameterize the control rates. However, training SNNs with parametric control rates is a brand-new challenge since there is almost no mature experience of training hyper-parameters in neural networks to borrow. The difficulty is twofold: (1) The existing SNNs are almost trained based on the spike response model scheme. This leads to the membrane potential in Eq. (2) is dominated by an indirect product interaction of connection weights $\mathbf{W}_j$ and control rate γ. So we cannot optimize the control rates and connection weights in parallel. (2) The roles of control rates and connection weights are distinctly different; each control rate is convolved with the received spikes aggregated by connection weights. So the spike errors caused by control rates spread temporally, while connection weights only transmit errors between layers. Formally, let E be the empirical loss, then we have

$$\frac{\partial E}{\partial \gamma_k} = \frac{\partial E}{\partial u_k}\frac{\partial u_k}{\partial \gamma_k} = \frac{\partial E}{\partial u_k}\left(\sum_{j=1}^{M}\mathbf{W}_j\left[\int_{t'}^{t}\exp\left(\gamma_k(s-t')\right)(s-t')I_j(s)\,\mathrm{d}s\right]\right),$$

where the subscript k denotes the k-th spiking neuron. When the firing period of the concerned spiking neuron is too small, *i.e.*, $(t-t') \to 0$, the gradients appear to vanish as $\partial u_k/\partial \gamma_k \to 0$; when the firing period is slightly longer, *i.e.*, $t \gg t'$, the gradients will be of explosion though time. To sum up, simply utilizing gradient-based methods to train the control rates is easier to fall into the quagmire of gradient vanishing and explosion.

An alternative approach to alleviate the issue is to employ alternating optimization for estimating control rate hyper-parameters. The key idea is to regard the control rates as a group of hyper-parameters drawn from a prior distribution so that solving for each learnable variable (connection weights or control rates) reduces to well-known methods. More formally, we consider a fully-connected feed-forward architecture and list this optimization procedure as follows:

Initialization: Sampling a group of control rates $\boldsymbol{\gamma} = \{\gamma_k\}$ from a pre-given distribution, such as the uniform distribution $\mathcal{U}[-1, 1]$. Then spikes spread according to

$$\begin{cases} S_j^1(t) = I_j(t), \\ u_k^{l+1}(t) = \sum_{j=1}^{n_l} \mathbf{W}_{kj}^l \left[\int_{t'}^{t} \exp\left(\gamma_k(s - t')\right) S_j^l(s) \, \mathrm{d}s \right], \\ S_k^{l+1}(t) \leftarrow f_e\left(u_k^{l+1}(t)\right), \end{cases}$$

where n_l is the number of spiking neurons in the l-th layer, $\mathbf{W}$ is the connection weight matrix, the superscript l denotes the l-th layer, and the subscripts k and j indicate the k-th and j-th spiking neurons, respectively.

Update connection weights: How to update the connection weights $\mathbf{W}$ with fixed $\boldsymbol{\gamma}$ depends on the choice of error-propagation techniques. Here, we employ a considerably mature work, SLAYER (Shrestha and Orchard, 2018) as a basic model.

Update $\boldsymbol{\gamma}$: We solve $\arg\min_{\boldsymbol{\gamma}} \mathrm{Loss}(\boldsymbol{\gamma} | \boldsymbol{I}, \boldsymbol{y}; \mathbf{W})$, where $\boldsymbol{y}$ denotes the supervised signals. Thus, existing algorithms, such as alternating coordinate descent (Beck and Tetruashvili, 2013) or Bayesian optimization (Snoek et al., 2012; Zhang and Zhou, 2020) can be applied directly to find a collection of apposite control rates.

Notice that the approaches based on alternating optimization place larger demands on computation and storage, and usually converge slowly in neural networks.

4.3 Dynamical Properties for Other Spiking Neural Models

Adhering to the analysis framework mentioned in Subsection 4.1, we here introduce several famous spiking neural models and analyze their dynamical properties. The first concerned one is the HH model (Hodgkin and Huxley, 1952), which usually be described as follows:

$$\begin{cases} \tau_u \frac{\mathrm{d}u}{\mathrm{d}t} = -g_1 u - g_2 u m^3 h - g_3 u n^4 + f(\boldsymbol{I}(t)), \\ \tau_n \frac{\mathrm{d}n}{\mathrm{d}t} = -n, \\ \tau_m \frac{\mathrm{d}m}{\mathrm{d}t} = -m, \\ \tau_h \frac{\mathrm{d}h}{\mathrm{d}t} = -h, \end{cases}$$

where u is the concerned electro-chemical membrane variable, n, m, and h describe the opening and closing of the voltage-dependent channels, τ_u is the capacitance of the membrane u, τ_n, τ_m, and τ_h are the corresponding membrane time hyper-parameters, g_1, g_2, and g_3 denote the conductance parameters for the different ion channels (*e.g.*, sodium Na and potassium K), and f represents the aggregation function of input signals $\boldsymbol{I}(t)$, as mentioned in Section 3.

Corollary 1 *The HH model with positive hyper-parameters (*i.e.*, g_1, τ_u, τ_n, τ_m, and τ_h) is a dissipative system.*

Proof The energy function of the HH model can be given by $\mathcal{H}_{\mathrm{HH}}(u,n,m,h,t) = |\boldsymbol{u}|^2 + o(|\boldsymbol{u}|) + 2F(u,t)$, where $\boldsymbol{u} = (u,n,m,h)$ is a short notation and $o(|\boldsymbol{u}|) = -2(g_2 m^3 h + g_3 n^4 h)u^2$ denotes the high-order term of $|\boldsymbol{u}|$, and F is the primitive function of $f \cdot u$, representing the non-potential forces. Thus, we can obtain its time derivative

$$\frac{\mathrm{d}\mathcal{H}_{\mathrm{HH}}}{\mathrm{d}t} = -2\left(g_1\frac{u^2}{\tau_u} + \frac{n^2}{\tau_n} + \frac{m^2}{\tau_m} + \frac{h^2}{\tau_h}\right).$$

Therefore, we have $\mathrm{d}\mathcal{H}_{\mathrm{HH}}/\,\mathrm{d}t < 0$ since $g_1, \tau_u, \tau_n, \tau_m, \tau_h > 0$. This completes the proof. ■

We also investigate Izhikevich's neuron model (Izhikevich, 2003), which is a good compromise between biophysical plausibility and computational cost. It is usually simple formulated by the following coupled equations

$$\begin{cases} \dfrac{\mathrm{d}u}{\mathrm{d}t} = a_u u^2 + b_u u - w + f(\boldsymbol{I}(t)), \\ \dfrac{\mathrm{d}w}{\mathrm{d}t} = a_w(b_w u - w), \end{cases}$$

where u and w are dimensionless membrane variables, a_u, b_u, a_w, and b_w are (positive) dimensionless hyper-parameters, and f represents the corresponding aggregation function of input signals $\boldsymbol{I}(t)$. Izhikevich's neuron model usually employs a group of typical values, *i.e.*, $a_u = 0.04$, $b_u = 5$, $a_w = 0.02$, and $b_w = 0.2$.

Corollary 2 *Izhikevich's neuron model leads to a hyperbolic system.*

Proof Write its energy function $\mathcal{H}_{\mathrm{I}}(u,w,t) = |(u,w)|^2 + o(|u|) + 2F(u,t)$, where $o(|u|) = 2a_u u^3$ denotes the high-order term of $|u|$ and F is the primitive function of $f \cdot u$, representing the non-potential forces. Then the time derivative is at your fingertips

$$\frac{\mathrm{d}\mathcal{H}_{\mathrm{I}}}{\mathrm{d}t} = 2(u,w)\mathbf{A}(u,w)^{\top} \quad \text{with} \quad \mathbf{A} = \begin{pmatrix} b_u & 0 \\ a_w b_w & -a_w \end{pmatrix}.$$

Izhikevich's neuron model has more complicated dynamical properties. Since the matrix $\mathbf{A}$ has two sign-opposite eigenvalues, *i.e.*, $\rho_1 = b_u > 0$ and $\rho_2 = -a_w < 0$, the tangent space becomes a hyperbolic manifold. So the neuron has two fixed points: a saddle point and an attractor. Around the saddle point, the dynamical system is unstable; the membrane potential will not stay at this point and the neuron either tends to fire or returns to near 0. However, the dynamical system around the attractor is relatively stable. Once the membrane potential trap in this region, it is difficult for neurons to fire again. This completes the proof. ■

5. Bifurcation Spiking Neural Networks

We present the BSNN for achieving adaptive SNN systems. In contrast to the LIF model that the eigenvalue is proportional to its control rate, BSNN employs a group of trainable parameters to separate the eigenvalues of the spiking neuron model from the control rates, making it possible to achieve diverse and learnable eigenvalues. Formally, we present the bifurcation spiking neurons model as follows:

$$\frac{\partial \boldsymbol{u}(t)}{\partial t} = \gamma \boldsymbol{u}(t) + \boldsymbol{\lambda} \boldsymbol{u}^* + \frac{R}{\tau_m} \boldsymbol{f}(\boldsymbol{I}), \tag{8}$$

where γ is the control rate, $\boldsymbol{\lambda}$ is a bifurcation parameter matrix, and the vector $\boldsymbol{u}^* = (u_1^*, \ldots, u_N^*)$ portrays the mutual promotion between neurons. Here, we unfold the k-th variable as $u_k^* = \sum_{i \neq k} u_i + o(|u_k|)$, where $o(|u_k|)$ denotes the high-order term of u_k. Thus, Eq. (8) becomes

$$\begin{cases} \dfrac{\partial u_1(t)}{\partial t} = \gamma\, u_1(t) + \displaystyle\sum_{i \neq 1} \lambda_{1i} u_i + o(|u_1|) + \frac{R}{\tau_m} f_1(\boldsymbol{I}), \\ \quad \vdots \qquad\qquad\qquad \vdots \\ \dfrac{\partial u_N(t)}{\partial t} = \gamma\, u_N(t) + \displaystyle\sum_{i \neq N} \lambda_{Ni} u_i + o(|u_N|) + \frac{R}{\tau_m} f_N(\boldsymbol{I}), \end{cases}$$

where

$$f_k(\boldsymbol{I}) = \sum_{j=1}^{M} \mathbf{W}_{kj} I_j(t).$$

The basic building block of BSNN is a system of equations concerning a cluster of spiking neurons. Invoke this cluster of spiking neurons to constitute a layer and reuse Eq. (8) layer by layer. Then we can establish a feed-forward multi-layer network.

5.1 Adaptive Dynamical System

To ensure BSNN becomes an adaptive dynamical system, we need to verify that the time derivative of the energy function produced by BSNN has learnable and diverse eigenvalues. Similar to the analysis in Section 4, we have the total energy of the BSNN model as follows

$$\mathcal{H}_{\mathrm{B}}(\boldsymbol{u}, t) = |\boldsymbol{u}|^2 + \frac{2R}{\tau_m} F(\boldsymbol{u}, t).$$

Correspondingly, the time derivative of energy function can be calculated by

$$\frac{\mathrm{d}\mathcal{H}_{\mathrm{B}}}{\mathrm{d}t} = 2\boldsymbol{u}^\top L_{\boldsymbol{\lambda}} \boldsymbol{u} \quad \text{with} \quad L_{\boldsymbol{\lambda}} = \begin{pmatrix} \gamma & & \\ & \ddots & \\ & & \gamma \end{pmatrix} + \boldsymbol{\lambda}.$$

Further, we can declare that BSNN has non-trivial solutions for achieving adaptive dynamical system.

Theorem 2 *If the bifurcation hyper-parameters λ_{ij} are all great than 0, there are at most 2^{N-1} bifurcation solutions in Eq.* (8)*.*

Proof Theorem 2 can be roughly proved by the following steps. First, finding the characteristic roots of our proposed BSNN model. According to Eq. (8), we can obtain its algebraic representation

$$\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}t} = L_{\boldsymbol{\lambda}} \boldsymbol{u} + G(\boldsymbol{u}, \boldsymbol{\lambda}) \quad \text{with} \quad L_{\boldsymbol{\lambda}} = A + B_{\boldsymbol{\lambda}} \quad \text{and} \quad G(\boldsymbol{u}, \boldsymbol{\lambda}) = o(|\boldsymbol{u}|),$$

where

$$A = \begin{pmatrix} \gamma & & \\ & \ddots & \\ & & \gamma \end{pmatrix} \quad \text{and} \quad B_{\boldsymbol{\lambda}} = \begin{pmatrix} 0 & \lambda_{12} & \dots & \lambda_{1n} \\ \lambda_{21} & 0 & \dots & \lambda_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \lambda_{n1} & \lambda_{n(n-1)} & \dots & 0 \end{pmatrix}.$$

Suppose that the eigenvalues of the matrix $B_{\boldsymbol{\lambda}}$ are $\beta_1, \dots, \beta_n$. So the eigenvalue ρ_i of $L_{\boldsymbol{\lambda}}$ can be calculated as the sum of that of A and that of $B_{\boldsymbol{\lambda}}$, that is, $\rho_i = -\gamma + \beta_i$.

Next, we can elucidate the bifurcation solutions relative to the eigenvalues. For simplicity, we take the 2-neuron model as an example, that is,

$$A = \begin{pmatrix} \gamma & 0 \\ 0 & \gamma \end{pmatrix} \quad \text{and} \quad B_{\boldsymbol{\lambda}} = \begin{pmatrix} 0 & \lambda_1 \\ \lambda_2 & 0 \end{pmatrix}.$$

Let $D_1 = 2\gamma$ and $D_2 = \gamma^2 - \lambda_1\lambda_2$. When $\Delta = D_1^2 - 4D_2 = \lambda_1\lambda_2 \geq 0$, $L_{\boldsymbol{\lambda}}$ has two real eigenvalues

$$\rho_1 = \frac{-D_1 - \sqrt{D_1^2 - 4D_2}}{2} \quad \text{and} \quad \rho_2 = \frac{-D_1 + \sqrt{D_1^2 - 4D_2}}{2}.$$

Obviously, ρ_1 must be less than zero, whereas ρ_2 is indefinite. Let $\lambda_c = \gamma^2$ be the critical threshold, then the bifurcation solutions of Eq. (8) are dominated by one pair of bifurcation eigenvalues

$$\begin{cases} \rho_1 = \dfrac{-D_1 - \sqrt{D_1^2 - 4D_2}}{2} < 0, \\ \rho_2 = \dfrac{-D_1 + \sqrt{D_1^2 - 4D_2}}{2} \begin{cases} < 0, & \lambda_1\lambda_2 < \lambda_c; \\ = 0, & \lambda_1\lambda_2 = \lambda_c; \\ > 0, & \lambda_1\lambda_2 > \lambda_c. \end{cases} \end{cases}$$

As long as the product of λ_1 and λ_2 is greater than 0, there exists at least one non-trivial solution of Eq. (8). In detail, when $\lambda_1\lambda_2 < \lambda_c$, both eigenvalues are negative, and thus, the whole dynamical system consists of two dissipative sub-systems, where both spiking neurons tend to hinder spike excitation. When $\lambda_1\lambda_2 = \lambda_c$, the whole dynamical system consists of a dissipative sub-system relative to the negative eigenvalue ρ_1 and a conservative one that ρ_2 is equal to 0. When $\lambda_1\lambda_2 > \lambda_c$, a new bifurcation phenomenon occurs, the whole dynamical system becomes structurally unstable, intuitively, one neuron still works in a "leaky" mode, but the other one contributes to spike excitation.

The existence of bifurcation eigenvalues is equivalent to the existence of non-trivial solutions of Eq. (8), and one pair of bifurcation solutions induces a group of apposite eigenvalues for achieving adaptive dynamical systems. Generally, for the case of N neurons, the solution of Eq. (8) possesses at most 2^{N-1} bifurcation solutions. ■

Based on the results of Theorem 2, the eigenvalues of Eq. (8) are dominated by a series of bifurcation parameters $\boldsymbol{\lambda}$. So we can convert the issue of searching for apposite control rate hyper-parameters into a new problem of how to train the bifurcation parameters. Therefore, even if the control rate is fixed, BSNN can still achieve an adaptive dynamic system. The training procedure for BSNN is introduced in the rest of this section.

5.2 Implementation

Here, we consider a feed-forward BSNN with M pre-synaptic input channels and N-dimensional output spiking neurons, and approximate the mutual promotion from the i-th neuron to the k-th neuron using the last spike of neuron i, noted as $S_i(t_i')$, where $t_i' = \max\{s|u_i(s) = u_{firing}, s < t_i\}$. Formally, for the k-th neuron, we have

$$\frac{\mathrm{d}u_k(t)}{\mathrm{d}t} = \gamma u_k(t) + \sum_{i=1,i\neq k}^{N} \lambda_{ki} S_i(t_i') + \sum_{j=1}^{M} \mathbf{W}_{kj} I_j(t). \tag{9}$$

According to Eq. (9), BSNN has two types of learnable parameters, *i.e.*, bifurcation parameters $\boldsymbol{\lambda}$ and connection weights $\mathbf{W}$, where $\boldsymbol{\lambda}$ is linearly independent to $\mathbf{W}$ at time t. Thus, BSNN avoids the problem of parameters entanglement.

Akin to the spike response model scheme (Gerstner, 1995), Eq. (9) has a closed-form solution

$$u_k(t) = \int_{t'}^{t} \exp\big(\gamma(s - t')\big) \cdot Q_k(s)\,\mathrm{d}s \quad \text{with} \quad Q_k(t) = \sum_{j=1}^{M} \mathbf{W}_{kj} I_j(t) + \sum_{i\neq k} \lambda_{ki} S_i(t_i'). \tag{10}$$

Finally, the generated spike is transmitted to the next neuron via the spike excitation function f_e.

5.3 Error Backpropagation in BSNN

BSNN with supervised signals can also be optimized via error backpropagation. Firstly, we denote the input (spike) sequence to a neuron as the following general form (Huh and Sejnowski, 2018)

$$I_j(t) = \sum_{firing} \epsilon_j \left(t - t_j^{firing}\right),$$

where t_j^{firing} is the spike time of the j-th input and $\epsilon(t)$ is a corresponding Dirac-delta function. Summing up the loss of the k-th target supervised signal $\hat{S_k}(t)$ related to $S_k(t)$ in time interval $[0, T]$

$$E_k = \frac{1}{2} \int_0^T E_k(t)\,\mathrm{d}t = \frac{1}{2} \int_0^T \left(S_k(t) - \hat{S_k}(t)\right)^2 \mathrm{d}t. \tag{11}$$

So for time t, we have

$$\frac{\partial E_k(t)}{\partial \mathbf{W}_{kj}} = \frac{\partial E_k(t)}{\partial S_k} \frac{\partial S_k}{\partial u_k} \frac{\partial u_k}{\partial \mathbf{W}_{kj}}, \tag{12}$$

where the first term is the error back-propagation of the excitatory neurons, the second term is that of the generated spikes with respect to the membrane potential, and the third term denotes the that of basic bifurcation neuron. Plugging Eqs. (10) and (11) into Eq. (12), we have

$$\frac{\partial E_k(t)}{\partial \mathbf{W}_{kj}} = \left(S_k(t) - \hat{S_k}(t)\right) f_e'(u_k) \delta_j(t) \quad \text{with} \quad \delta_j(t) = \frac{\epsilon_j(t)}{\tau_m} \exp(\gamma t).$$

However, the derivative of the spike excitation function $f_e'(u)$ is a persistent problem for training SNNs with supervised signals. Recently, there have emerged many seminal approaches for addressing this problem. In this paper, we directly employ the result of Shrestha and Orchard (2018). Therefore, we obtain the backpropagation pipeline related to connection weights $\mathbf{W}_{kj}$

$$\nabla_{\mathbf{W}_{kj}} E = \int_0^T \frac{\partial E_k(t)}{\partial \mathbf{W}_{kj}} \, \mathrm{d}t.$$

Similarly, the correction formula with respect to λ_{ki} is given by

$$\nabla_{\lambda_{ki}} E = \int_0^T \left(\hat{S}_k(t) - S_k(t) \right) f_e'(u_k) S_i(t_i') \gamma \exp\left(\gamma t\right) \mathrm{d}t.$$

We can also add a learning rate to help convergence, just like most deep artificial neural networks.

Here, BSNN is implemented by an extended backpropagation algorithm. Compared with the existing SNNs, BSNN only needs to calculate one more set of gradients, *i.e.*, $\nabla_{\lambda_{ki}} E$ during feedback. The records of $S_i(t_i')$ do not cause additional storage because we intrinsically need the membrane potential values of each spiking neuron during the gradient calculation procedure as shown in Eq. (12). So both the computation and storage of BSNN are considerably less in comparison with the alternating optimization approaches.

In summary, the proposed BSNN has at least three advantages: (1) BSNN employs learnable parameters to adjust eigenvalues, and thus, its topology becomes flexible, enabling SNNs to capture the adaptive learning behavior of spiking neurons to the environment change. (2) Since the new-added bifurcation parameters $\boldsymbol{\lambda}$ are independent of the connection weights $\mathbf{W}$, these two will not be entangled and conflict when using the spike response model scheme to train SNNs. (3) The performance of BSNN depends less on the control rates.

6. Experiments

In this section, we conducted experiments on four benchmark data sets to evaluate the functional performance of BSNN. The experiments are performed to discuss the following questions:

Q1: Is the performance of BSNN comparable with state-of-the-art SNNs?

Q2: Does the performance of BSNN surpass that of alternating optimization, especially in terms of accuracy and efficiency?

Q3: Concerning BSNN, is the performance robust to the control rate? In which conditions?

Data Sets: (1) The MNIST handwritten digit data set[1] comprises a training set of 60,000 examples and a testing set of 10,000 examples in 10 classes, where each example is centered in a 28×28 image. Using Poisson encoding, we produce a list of spike signals with a formation of $784 \times T$ binary matrices, where T denotes the encoding length and each row represents a spike sequence at each pixel. (2) The Neuromorphic-MNIST (N-MNIST) data set[2] (Orchard et al., 2015) is a spiking version of the original frame-based MNIST data set. Each example in N-MNIST was converted into a spike sequence by mounting the ATIS sensor on a motorized pan-tilt unit and having

1. http://yann.lecun.com/exdb/mnist/
2. https://www.garrickorchard.com/datasets/n-mnist

Table 1: Parameter Setting of BSNN on Various Data Sets.

Parameters Value	**MNIST**	**N-MNIST**	**Fashion-MNIST**	**EMNIST**
Batch Size	32	32	32	64
Encoding Length T	300	300	400	400
Expect Spike Count(True)	100	80	100	140
Expect Spike Count(False)	10	5	10	0
Learning Rate η	0.01	0.01	0.001	0.01
Refractory Period	2 ms	1 ms	2 ms	2 ms
Time Constant of Synapse τ_s	8 ms	8 ms	8 ms	8 ms

the sensor move while it views MNIST examples on an LCD monitor. It consists of the same 60,000 training and 10,000 testing samples as the original MNIST data set, and is captured at the same visual scale as the original MNIST data set (28 $\times$ 28 pixels) with both "on" and "off" spikes. (3) The Fashion-MNIST data set[3] consists of a training set of 60,000 examples and a testing set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. (4) The Extended MNIST-Balanced (EMNIST) (Cohen et al., 2017) data set is an extension of MNIST to handwritten, which contains handwritten upper & lower case letters of the English alphabet in addition to the digits, and comprises 112,800 training and 18,800 testing samples for 47 classes.

Customization: The pre-processing steps for these experiments are the same as those used by Pillow et al. (2005); Quiroga et al. (2005); Anumula et al. (2018). Each static image of (1) MNIST, (3) Fashion-MNIST, and (4) EMNIST is transformed as a spike sequence using Poisson Encoding, while each instance in N-MNIST was encoded by a Dynamic Audio / Vision Sensor (DAS / DVS). For these image classification tasks, we set 10 output spiking neurons corresponding to the classification labels. The output label of SNNs is the one with the greatest spike count. Table 1 lists the typical constant values in BSNNs.

Contenders: We also employ two types of contenders to competing with the proposed BSNN: (1) several state-of-the-art SNNs with the spike response model scheme and (2) alternating optimization algorithms, see Subsection 4.2. In this work, all SNN models are without any convolution structure. The alternating optimization algorithms pre-sample a group of control rates from two uniform distributions, $U_1 = \mathcal{U}[-1, 0]$ and $U_2 = \mathcal{U}[-1, 1]$. For example, SLAYER-U_1 denotes an alternating optimization method based on SLAYER, which draws control rate hyper-parameters from $\mathcal{U}[-1, 0]$.

Experimental Results: Table 2 lists the comparative performance (accuracy) and configurations (setting and epoches) of the contenders and BSNN on 3 digit data sets. As we can see, BSNN performs best against other competing approaches, achieving very superior testing accuracy (*i.e.*, more than 99% on MNIST, around 99.24% on NMNIST, and more than 91% on Fashion-MNIST). It is a laudable result for SNNs. In addition, the alternating optimization algorithms, *i.e.*, both SLAYER-U_1 and SLAYER-U_2 steadily surpass the original SLAYER algorithm, which demonstrates the way of achieving diverse and adaptive control rates is significant and effective for improving the performance of SNNs.

To help understand the learning process of SNNs and highlight the difference between the contenders and BSNN, we illustrate the spike raster plots of SLAYER, SLAYER-U_2, and BSNN. We show the neuron excitation snapshots of these three approaches on the 4881-st and 6429-th MNIST testing samples with label 0 in Figure 2. The horizontal and vertical axes indicate the time interval

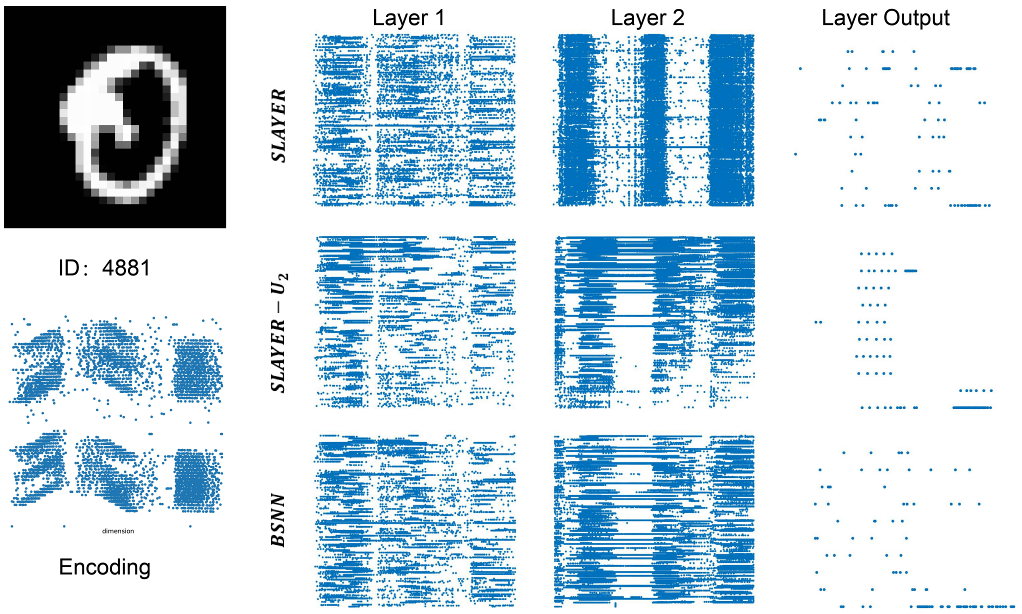
3. https://www.kaggle.com/zalando-research/fashionmnist

Table 2: The comparative performance of the contenders and BSNN.

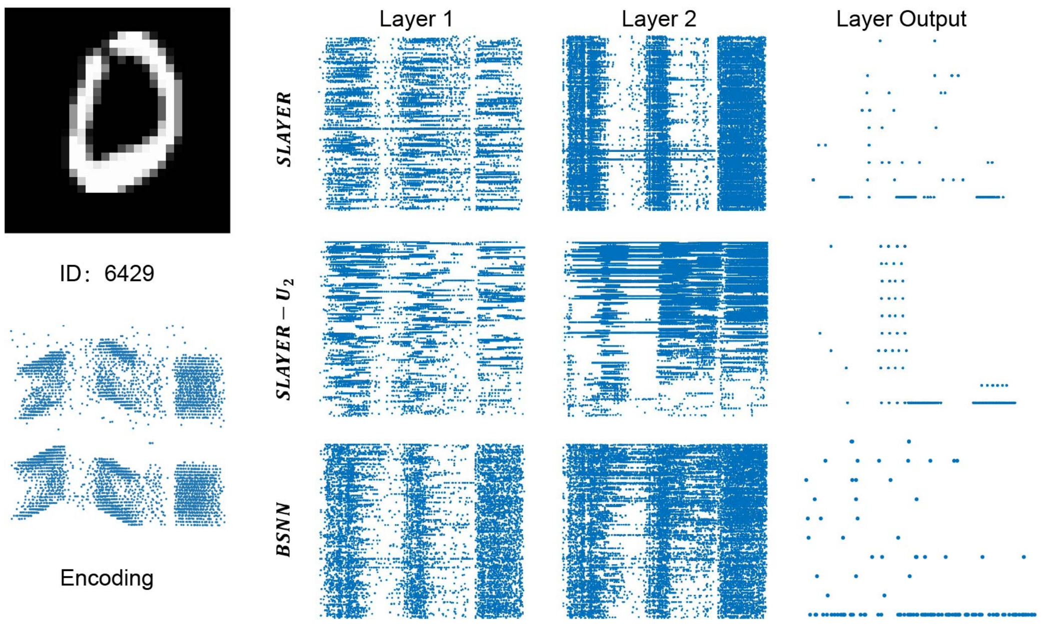
Data Sets	Contenders	Accuracy (%)	Setting	Control Rate (γ)	Epochs
MNIST	Deep SNN (O'Connor and Welling, 2016)	97.80	28×28-300-300-10 ♠	-	50
	Deep SNN-BP (Lee et al., 2016)	98.71	28×28-800-10	-	200
	SNN-EP ♡	97.63	28×28-500-10	-	25
	HM2-BP (Jin et al., 2018)	98.84 ± 0.02	28×28-800-10	-	100
	SNN-L (Rezaabad and Vishwanath, 2020)	98.23 ± 0.07	28×28-1000-R28-10	-	-
	SLAYER (Shrestha and Orchard, 2018)	98.39 ± 0.04	28×28-500-500-10	-	50
	SLAYER-U_1 ♣	98.53 ± 0.03	28×28-500-500-10	-	-
	SLAYER-U_2	98.59 ± 0.01	28×28-500-500-10	-	-
	BSNN (this work)	**99.02 ± 0.04**	28×28-500-500-10	-0.21	50
N-MNIST	SKIM (Cohen et al., 2016)	92.87	2*28×28-10000-10	-	-
	Deep SNN-BP	98.78	2*28×28-800-10	-	200
	HM2-BP	98.84 ± 0.02	2*28×28-800-10	-	60
	SLAYER	98.89 ± 0.06	2*28×28-500-500-10	-	50
	SLAYER-U_1	99.01 ± 0.01	2*28×28-500-500-10	-	-
	SLAYER-U_2	99.07 ± 0.02	2*28×28-500-500-10	-	-
	BSNN (this work)	**99.24 ± 0.12**	2*28×28-500-500-10	-0.49	50
Fashion-MNIST	HM2-BP	88.99	28×28-400-400-10	-	15
	SLAYER	88.61 ± 0.17	28×28-500-500-10	-	50
	SLAYER-U_1	90.53 ± 0.04	28×28-500-500-10	-	-
	SLAYER-U_2	90.61 ± 0.02	28×28-500-500-10	-	-
	ST-RSBP (Zhang and Li, 2019)	90.00 ± 0.13	28×28-400-R400-10 ◇	-	30
	BSNN (this work)	**91.22 ± 0.06**	28×28-500-500-10	-0.32	50
EMNIST	eRBP (Neftci et al., 2017)	78.17	28×28-200-200-47	-	30
	HM2-BP	84.43 ± 0.10	28×28-400-400-10	-	20
	SNN-L	83.75 ± 0.15	28×28-1000-R28-10	-	-
	SLAYER	85.73 ± 0.16	28×28-500-500-47	-	50
	SLAYER-U_2	86.62 ± 0.03	28×28-500-500-47	-	50
	BSNN (this work)	**87.51 ± 0.23**	28×28-500-500-47	-0.37	50

♠ :-300-300- denotes two hidden layers with 300 spiking neurons, while -800- is one hidden layer with 800 spiking neurons.
♡ : SNN-EP (O'Connor et al., 2019) proposes an implementation for training SNN with equilibrium propagation.
♣ : -U_1 and -U_2 indicate the alternating optimization algorithms with parametric control rates sampled from U_1 and U_2, respectively.
◇ : R400 represents a recurrent layer of 400 spiking neurons.

and the sequence of spiking neurons or encoding channels, respectively. In detail, we pick up a representative instance, the 4881-st one, and then convert this image into a spike sequence using a DVS, as shown in the left two subplots of Figure 2(a). It is observed that there are two fractures, one around time interval $[80, 100]$ and the other around time interval $[200, 230]$. The spike raster plots of spiking neurons (in Layer 1, Layer 2, and Layer Output) of SLAYER, SLAYER-U_2, and BSNN are successively shown in the right nine subplots. As mentioned in Section 4, a dissipating system would hinder the spike excitation. Thus, we can observe the plots of SLAYER with the following properties. (1) The neurons in the same hidden layer have almost the same firing rates. (2) The plots of Layer 1 and Layer 2 of SLAYER display two distinct fractures. Specifically, Layer 2 even enlarges the fracture. In practice, it is more likely to fall into the situation of "dead neurons". (3) In Layer Output, the neuron corresponding to label 0 has no obvious advantage over other neurons in terms of total spiking counts, leading this instance to be incorrectly classified as label 8. In contrast, both SLAYER-U_2 and BSNN have indefinite eigenvalues, where negative ones hinder the spike excitation, positive ones promote the spike excitation, and ones with zero are a conservative system. Thus, the developed models can dynamically determine the firing rates of spiking neurons. In detail, (1) the firing rates of spiking neurons in the same layer show significant differences. (2) The output spiking



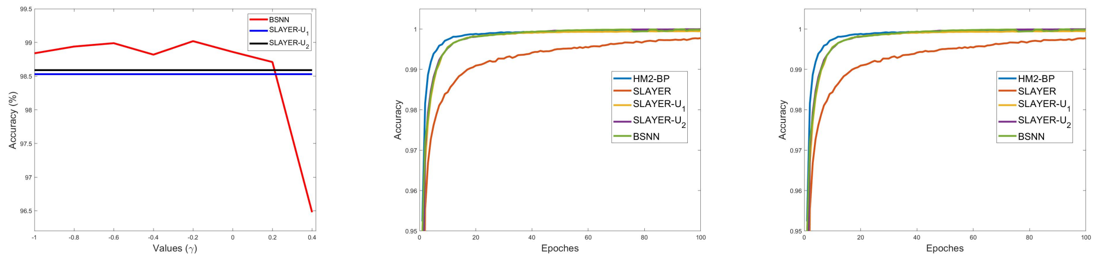
(a) Example 4881.



(b) Example 6429.

Figure 2: The spike raster plots of SLAYER, SLAYER-U_2, and BSNN on (a) 4881 and (b) 6429.

neurons relative to wrong labels are suppressed. In contrast, the neuron relative to the correct label is "encouraged" to fire more spikes and eventually win with a significant advantage.



(a) (b) (c)

Figure 3: (a) Robustness of BSNN with respect to control rates. The training (b) and testing (c) accuracy curves of the contenders and BSNN on MNIST.

We also demonstrate the robustness of BSNN to the control rate. This experiment is conducted on the MNIST data set, setting the architecture of BSNN as 28×28-500-500-10. The control rate hyper-parameters are optimized by grid search. For each control rate value, we ran BSNN 5 times, recorded the largest accuracy of each round within 50 epochs, and averaged five accuracy records as its performance. The results plotted in Figure 3(a) show that BSNN is able to perform better than the alternating optimization algorithms in a broad-range setting of control rates. Besides, Figure 3(b) and (c) show the learning curves with $\gamma = -0.21$ on MNIST. As we can see, BSNN converges fast and surpasses the contenders.

Based on the experimental results and discussions above, we can conclude that BSNN achieves superior performance to the existing SNNs and the improved contenders - approaches based on alternating optimization algorithms. Additionally, the performance of BSNN is considerably robust to the setting of control rates.

7. Conclusion and Prospects

In this paper, we investigated spiking neural models and networks from the perspective of dynamical systems. We reveal the dynamical properties of spiking neural models by providing their energy function and conclude that the LIF model is a bifurcation dynamical system, where the control rates are the corresponding bifurcation hyper-parameters. Further, by employing the spiking neural model to enable adaptable eigenvalues, we proposed the *Bifurcation Spiking Neural Network* (BSNN). Compared with the alternating optimization approaches, BSNN tackles the challenge that control rates interact with connection weights in the training procedure, leading to a robust setting of control rates. Besides, BSNN achieves a better accuracy with considerably less computation and storage in supervised classification tasks. The experiments verified the effectiveness of BSNN.

We introduced a mathematical framework for analyzing the spiking neural models, including but not limited to using a Lyapunov-like function to formulate the total energy, revealing the bifurcation characteristics of SNNs, and providing systematic guidance on hyper-parameter setting in SNNs. These results may contribute to the development of SNN-related theories. Besides, we declare that our work doesn't aim to realize a biological learning phenomenon but to explore some new thoughts on SNNs. In this situation, Eq. (9) that employs the last spikes of adjacent neurons to approximate the mutual promotion only provides a feasible paradigm of implementing dynamic bifurcation neurons.

There are some important future issues. First, it is crucial to develop in-depth theoretical understandings of SNNs, such as from aspects of approximation complexity (Eldan and Shamir, 2016), algorithmic power (Barrett et al., 2013; Chou et al., 2019), representation ability (Zhang and Zhou, 2021a), and especially in over-parameterized architectures (Zhou, 2021). Second, recently another bio-plausible neuron model, the *Flexible Transmitter* (FT) model (Zhang and Zhou, 2021b), has been proposed. In contrast to the spiking neural model that is formulated as first-order parabolic equations, the FT neuron model mimics the neurotrophic potentiation and depression effects by a formulation of two-variable function, exhibiting great potential on temporal-spatial data processing. Much effort is required to explore the theoretical properties of the FT model (Wu et al., 2021), with a possible way from the perspective of dynamical systems.

Acknowledgments

This research was supported by the National Science Foundation of China (61921006) and the Collaborative Innovation Center of Novel Software Technology and Industrialization.

References

J. Anumula, D. Neil, T. Delbruck, and S.-C. Liu. Feature representations for neuromorphic audio spike streams. *Frontiers in Neuroscience*, 12:23, 2018.

D. G. Barrett, S. Denève, and C. K. Machens. Firing rate predictions in optimal balanced networks. In *Advances in Neural Information Processing Systems 26*, pages 1538–1546, 2013.

A. Beck and L. Tetruashvili. On the convergence of block coordinate descent type methods. *SIAM Journal on Optimization*, 23(4):2037–2060, 2013.

J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *The Journal of Machine Learning Research*, 13(1):281–305, 2012.

S. M. Bohte, J. M. Kok, and H. La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002.

A. N. Burkitt. A review of the integrate-and-fire neuron model: I. Homogeneous synaptic input. *Biological Cybernetics*, 95(1):1–19, 2006a.

A. N. Burkitt. A review of the integrate-and-fire neuron model: II. Inhomogeneous synaptic input and network properties. *Biological Cybernetics*, 95(2):97–112, 2006b.

K. D. Carlson, J. M. Nageswaran, N. Dutt, and J. L. Krichmar. An efficient automated parameter tuning framework for spiking neural networks. *Frontiers in Neuroscience*, 8:10, 2014.

C.-N. Chou, K.-M. Chung, and C.-J. Lu. On the algorithmic power of spiking neural networks. In *Proceedings of the 10th Innovations in Theoretical Computer Science Conference*, pages 26:1–26:20, 2019.

G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik. EMNIST: Extending MNIST to handwritten letters. In *Proceesdings of the 2017 International Joint Conference on Neural Networks*, pages 2921–2926, 2017.

G. K. Cohen, G. Orchard, S.-H. Leng, J. Tapson, R. B. Benosman, and A. Van Schaik. Skimming digits: Neuromorphic classification of spike-encoded images. *Frontiers in Neuroscience*, 10:184, 2016.

G. Dumont, A. Payeur, and A. Longtin. A stochastic-field description of finite-size spiking neural networks. *PLoS Computational Biology*, 13(8):1005691, 2017.

R. Eldan and O. Shamir. The power of depth for feedforward neural networks. In *Proceedings of 29th Annual Conference on Learning Theory*, pages 907–940, 2016.

W. Gerstner. Time structure of the activity in neural network models. *Physical Review E*, 51(1):738, 1995.

W. Gerstner and W. M. Kistler. *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge University Press, 2002.

M. W. Hirsch, S. Smale, and R. L. Devaney. *Differential Equations, Dynamical Systems, and An Introduction to Chaos*. Academic Press, 2012.

A. L. Hodgkin and A. F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500, 1952.

N. Hohn and A. N. Burkitt. Shot noise in the leaky integrate-and-fire neuron. *Physical Review E*, 63 (3):031902, 2001.

D. Huh and T. J. Sejnowski. Gradient descent for spiking neural networks. In *Advances in Neural Information Processing Systems 31*, pages 1440–1450, 2018.

E. Hunsberger and C. Eliasmith. Spiking deep networks with LIF neurons. *arXiv:1510.08829*, 2015.

D. Hunter, H. Yu, M. S. Pukish III, J. Kolbusz, and B. M. Wilamowski. Selection of proper neural network sizes and architectures — a comparative study. *IEEE Transactions on Industrial Informatics*, 8(2):228–240, 2012.

E. M. Izhikevich. Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, 14(6): 1569–1572, 2003.

Y. Jin, W. Zhang, and P. Li. Hybrid macro/micro level backpropagation for training deep spiking neural networks. In *Advances in Neural Information Processing Systems 31*, pages 7005–7015, 2018.

S. R. Kulkarni and B. Rajendran. Spiking neural networks for handwritten digit recognition—supervised learning and network optimization. *Neural Networks*, 103:109–127, 2018.

Y. A. Kuznetsov. *Elements of Applied Bifurcation Theory*. Springer, 2013.

J. H. Lee, T. Delbruck, and M. Pfeiffer. Training deep spiking neural networks using backpropagation. *Frontiers in Neuroscience*, 10:508, 2016.

J. L. Lobo, J. Del Ser, A. Bifet, and N. Kasabov. Spiking neural networks and online learning: An overview and perspectives. *Neural Networks*, 121:88–100, 2020.

P. R. Lorenzo, J. Nalepa, M. Kawulok, L. S. Ramos, and J. R. Pastor. Particle swarm optimization for hyper-parameter selection in deep neural networks. In *Proceedings of the 2017 Genetic and Evolutionary Computation Conference*, pages 481–488, 2017.

S. McKennoch, D. Liu, and L. G. Bushnell. Fast modifications of the spikeprop algorithm. In *Proceedings of the 2006 International Joint Conference on Neural Network*, pages 3970–3977, 2006.

E. O. Neftci, C. Augustine, S. Paul, and G. Detorakis. Event-driven random back-propagation: Enabling neuromorphic deep learning machines. *Frontiers in Neuroscience*, 11:324, 2017.

P. O'Connor and M. Welling. Deep spiking networks. *arXiv:1602.08323*, 2016.

P. O'Connor, D. Neil, S.-C. Liu, T. Delbruck, and M. Pfeiffer. Real-time classification and sensor fusion with a spiking deep belief network. *Frontiers in Neuroscience*, 7:178, 2013.

P. O'Connor, E. Gavves, and M. Welling. Training a spiking neural network with equilibrium propagation. In *Proceesdings of the 22nd International Conference on Artificial Intelligence and Statistics*, pages 1516–1523, 2019.

A. Onuki. *Phase Transition Dynamics*. Cambridge University Press, 2002.

G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor. Converting static image datasets to spiking neuromorphic datasets using saccades. *Frontiers in Neuroscience*, 9:437, 2015.

J. W. Pillow, L. Paninski, V. J. Uzzell, E. P. Simoncelli, and EJ Chichilnisky. Prediction and decoding of retinal ganglion cell responses with a probabilistic spiking model. *Journal of Neuroscience*, 25 (47):11003–11013, 2005.

R. Q. Quiroga, L. Reddy, G. Kreiman, C. Koch, and I. Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102, 2005.

A. Lotfi Rezaabad and S. Vishwanath. Long short-term memory spiking networks and their applications. In *Proceesdings of the 2020 International Conference on Neuromorphic Systems*, pages 1–9, 2020.

B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu. Conversion of continuous-valued deep networks to efficient event-driven networks for image classification. *Frontiers in Neuroscience*, 11:682, 2017.

T. Shimokawa, K. Pakdaman, and S. Sato. Time-scale matching in the response of a leaky integrate-and-fire neuron model to periodic stimulus with additive noise. *Physical Review E*, 59(3):3427, 1999.

S. B. Shrestha and G. Orchard. SLAYER: Spike layer error reassignment in time. In *Advances in Neural Information Processing Systems 31*, pages 1419–1428, 2018.

J. Snoek, H. Larochelle, and R. P. Adams. Practical bayesian optimization of machine learning algorithms. *Advances in Neural Information Processing Systems 25*, pages 2951–2959, 2012.

R. VanRullen, R. Guyonneau, and S. J. Thorpe. Spike times make sense. *Trends in Neurosciences*, 28(1):1–4, 2005.

J.-H. Wu, S.-Q. Zhang, Y. Jiang, and Z.-H. Zhou. Towards theoretical understanding of flexible transmitter networks via approximation and local minima. *arXiv:2111.06027*, 2021.

Q. Yu, H. Tang, K. C. Tan, and H. Yu. A brain-inspired spiking neural network model with temporal encoding and learning. *Neurocomputing*, 138:3–13, 2014.

S.-Q. Zhang and Z.-H. Zhou. Harmonic recurrent process for time series forecasting. In *Proceedings of the 24th European Conference on Artificial Intelligence*, pages 1714–1721, 2020.

S.-Q. Zhang and Z.-H. Zhou. Arise: Aperiodic semi-parametric process for efficient markets without periodogram and gaussianity assumptions. *arXiv:2111.06222*, 2021a.

S.-Q. Zhang and Z.-H. Zhou. Flexible transmitter network. *Neural Computation*, 33(11):2951–2970, 2021b.

W. Zhang and P. Li. Spike-train level backpropagation for training deep recurrent spiking neural networks. In *Advances in Neural Information Processing Systems 32*, pages 7800–7811, 2019.

Z.-H. Zhou. Why over-parameterization of deep neural networks does not overfit? *Science China Information Sciences*, 64(1):1–3, 2021.