Spiking Neural Networks with Laterally-Inhibited Self-Recurrent Units

Wenrui Zhang, Peng Li
Department of Electrical and Computer Engineering
University of California, Santa Barbara
Santa Barbara, CA, 93106, U.S.A.
wenruizhang@ucsb.edu, lip@ucsb.edu

***Abstract*—In biological brains, recurrent connections play a crucial role in cortical computation, modulation of network dynamics, and communication. However, in recurrent spiking neural networks (SNNs), recurrence is mostly constructed by random connections. How excitatory and inhibitory recurrent connections affect network responses and what kinds of connectivity benefit learning performance is still obscure. In this work, we propose a novel recurrent structure called the Laterally-Inhibited Self-Recurrent Unit (LISR), which consists of one excitatory neuron with a self-recurrent connection wired together with an inhibitory neuron through excitatory and inhibitory synapses. The self-recurrent connection of the excitatory neuron mitigates the information loss caused by the firing-and-resetting mechanism and maintains the long-term neuronal memory. The lateral inhibition from the inhibitory neuron to the corresponding excitatory neuron, on the one hand, adjusts the firing activity of the latter. On the other hand, it plays as a forget gate to clear the memory of the excitatory neuron. Based on speech and image datasets commonly used in neuromorphic computing, RSNNs based on the proposed LISR improve performance significantly by up to 9.26% over feedforward SNNs trained by a state-of-the-art backpropagation method with similar computational costs.**

***Index Terms*—Recurrent spiking neural networks, Spiking neural networks structure, Self-recurrent connections, Inhibition**

I. Introduction

Recurrence is ubiquitous in the brain and involved in most of the brain's dynamics. Recurrent connections between neurons play diverse functional roles for storing spatial patterns in memory [1], [2], winner-take-all decision making [3], [4], oscillations of multiple types [5], object recognition in the visual system [6], [7], and so on. Inspired by the connectivity in brain, recurrent connections have been widely applied in artificial neural networks (ANNs). During the past few decades, various structures of recurrent neural networks (RNNs) have been proposed such as Long Short Term Memory (LSTM) [8], Echo State Networks (ESN) [9], Deep RNNs [10], Gated Recurrent Units (GRU) [11], and Legendre Memory Units (LMU) [12].

As a brain-inspired computational model, spiking neural networks (SNNs) are considered as the third-generation of artificial neural networks with the more biologically realistic spiking neuron model. There is theoretical evidence supporting that SNNs possess greater computational power over traditional ANNs [13]. In ANNs, neurons process continuous-valued inputs with continuous outputs generated through an activation function. However, the spiking neurons mimic the biological neurons' behavior and explicitly model the all-or-none firing spikes across both spatial and temporal domains. Unlike recurrence in traditional ANNs that has been well studied with various structures proposed, the exploration of recurrent SNNs (RSNNs) is still immature due to the complex spatial-temporal dynamics.

Randomly connected recurrent layers or reservoirs are the most common structures for RSNNs. Among them, the Liquid State Machine (LSM) [14] is one of the most widely adopted architectures and has been demonstrated on tasks like speech recognition [15], [16], image classification [17], and so on. The LSM consists of a single randomly connected recurrent reservoir layer followed by one readout layer. Its recurrent weights are typically either fixed or trained by unsupervised learning methods like spike-timing-dependent plasticity (STDP) [18] with only the readout layer trained by supervision [15], [16], [19]. In recent years, the standard LSM has been extended to multiple reservoirs by applying different parts of the input signals to corresponding reservoirs [17], [20].

Apart from LSM, [21] proposed an architecture called long short-term memory SNNs (LSNNs). Its recurrent layer consists of a regular spiking portion with both inhibitory and excitatory spiking neurons and an adaptive neural population. The recurrent connections are trained by the backpropagation through time (BPTT) method. [22] demonstrates training of deep RSNNs by a backpropagation (BP) method called ST-RSBP. [23] proposed a recurrent structure in which the recurrent layer is organized in C clusters of excitatory neurons and a

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research under Award Number DE-SC0021319, and the National Science Foundation under Award Number 1948201.

central cluster of inhibitory neurons. The recurrent connections in all of these works are sparsely and randomly generated with certain probabilities. Whether a neuron is connected to another neuron in the same layer is determined by the probability. However, the randomly generated connections may not be optimal and thus limit the performance. In addition, the complex network dynamics created by the random recurrent connections also hinder the network training from learning tools and severely limit the practical application of RSNNs.

On the other hand, [24] proposed a recurrent structure named ScSr-SNNs. In its recurrent layer, each recurrent neuron only has a self-recurrent connection to the neuron itself. Although such a structure is easy to implement, its connections are so simple that it cannot exploit the full power of recurrence.

In this paper, we proposed a novel recurrent structure for SNNs called Laterally-Inhibited Self-Recurrent Unit (LISR). The LISR contains one excitatory neuron with a tunable self-recurrent connection wired together with an inhibitory neuron. In addition, the excitatory neuron has a fixed excitatory connection to the inhibitory neuron while the inhibitory neuron introduces lateral inhibition through a fixed inhibitory connection to the excitatory neuron. This proposed connectivity not only offers a structured approach for designing high-performance RSNNs but also mitigates the training challenges resulting from random recurrent connections as in the prior works. We evaluate the proposed structure trained by a state-of-the-art BP algorithm of SNNs on speech dataset TI46 [25], neuromorphic speech dataset N-TIDIGITS [26], and neuromorphic image dataset DVS-Gesture [27]. The networks with the proposed LISR outperform the best-reported performance obtained from the existing works on all of the three datasets. In addition, without increasing the computational cost, it achieves up to 9.26% performance improvement compared to the same size feedforward network trained by the BP method.

II. Methodology

A. Spiking Neuron Model

In this work, the leaky integrate-and-fire (LIF) neuron model [28], one of the most prevalent choices for describing dynamics of spiking neurons, is adopted.

The neuronal membrane voltage $u_p(t)$ of postsynaptic neuron p at time t is given by

$$\tau_m \frac{du_p(t)}{dt} = -u_p(t) + RI_p(t), \tag{1}$$

where R and τ_m are the effective leaky resistance and time constant of the membrane, $I_p(t)$ the integrated input current.

The neuron p is driven by the input current which is the weighted summation of postsynaptic current (PSC) from presynaptic neuron with the following general form:

$$I_p(t) = \sum_q w_{pq} a_q(t) \tag{2}$$

where w_{pq} is the synaptic weight from presynaptic neuron q to postsynaptic neuron p, and $a_q(t)$ the PSC induced by the spikes from neuron q.

The postsynaptic current (PSC) $a_q(t)$ is converted from the presynaptic spikes through a synaptic model. We adopt the first-order synaptic model [28] which is defined as

$$\tau_s \frac{da_q(t)}{dt} = -a_q(t) + s_q(t), \tag{3}$$

where τ_s is the synaptic time constant, $s_q(t)$ the spiking events of presynaptic neuron q. $s_q(t)$ can be expressed as

$$s_q(t) = \sum_{t_q^{(f)}} \delta(t - t_q^{(f)}), \tag{4}$$

where δ is the Dirac delta function and $t_q^{(f)}$ denotes the firing time of presynaptic neuron q.

During the simulation, we use the fixed-step first-order Euler method to discretize continuous membrane voltage updates into discrete time steps. Since the ratio of R and τ_m can be absorbed into the synaptic weights, (1) can be converted to

$$u_p[t] = \theta_m u_p[t-1](1 - s_p[t-1]) + \sum_q w_{pq} a_q[t], \tag{5}$$

where $\theta_m = 1 - \frac{1}{\tau_m}$ and the $1 - s_p[t-1]$ term reflects the effect of firing-and-resetting mechanism. The spiking neuron generates an output spike when $u_p[t]$ reaches the predetermined threshold V_{th} and reset the $u_p[t]$ to the rest potential.

B. Laterally-Inhibited Self-Recurrent Unit

In this part, we present the structure for the proposed Laterally-Inhibited Self-Recurrent Unit (LISR). It has clear and structured recurrent connections which can benefit the network in several aspects.

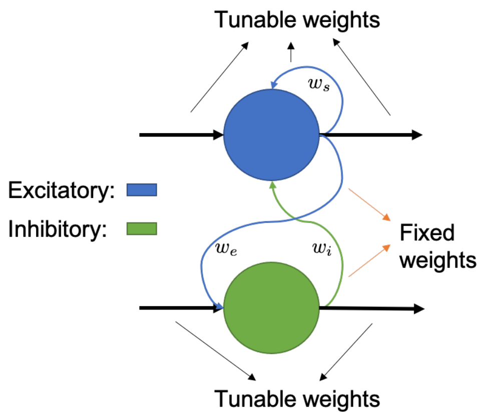

Fig. 1. Laterally-Inhibited Self-Recurrent unit.

In the recurrent layer, neurons are grouped into pairs. Each LISR consists of one excitatory neuron (E) with a self-recurrent connection and its corresponding inhibitory neuron (I). As shown in Fig. 1, there are three recurrent connections in a LISR, including a self-recurrent connection of E, an excitatory recurrent connection from E to I, and an inhibitory recurrent connection from I to E. Among these recurrent connections, the self-recurrent connection is trained by the learning rule while the weights of the other two connections are fixed. In addition, the neurons of two adjacent layers are fully connected. In this work, we suppose all the recurrent connections have delay of 1 time step and feedforward connections have no delay.

In the rest of this paper, we use e and i in the subscript to denote the variables of the excitatory neuron and inhibitory neuron, respectively. For the inhibitory neuron, by introducing the excitatory recurrent connection, the expression of the neuron model changes from (5) to

$$u_i[t] = \theta_m u_i[t-1](1 - s_i[t-1]) + I_i[t] + w_e a_e[t-1] \quad (6)$$

where w_e is the fixed weight of the excitatory connection and $a_e[t-1]$ the PSC of the excitatory neuron.

Similarly, the membrane potential of the excitatory neuron can be concluded as

$$u_e[t] = \theta_m u_e[t-1](1 - s_e[t-1]) + \sum_q w_{eq} a_q[t] + w_s a_e[t-1] + w_i a_i[t-1] \quad (7)$$

where w_s is the weight of self-recurrent connection, w_i the fixed weight of the inhibitory connection, and $a_i[t-1]$ the PSC of the inhibitory neuron.

In the proposed LISR, the excitatory neuron with self-recurrent connection plays a major role in information processing and feature extraction while receiving lateral inhibition from the inhibitory neuron. Thus, the following analysis is focused on the excitatory neuron.

[29] introduced the idea of connecting neurons back to themselves for ANNs. It claimed that a kind of longer-term memory can be formed by making part of the recurrent weight matrix close to the identity matrix. It is the same as adding self-recurrent connections to part of hidden neurons. After that, [30] proposed an independently recurrent neural network (IndRNN) for ANNs with self-recurrent connections. It demonstrated that multiple self-recurrent layers can be stacked to construct a deep network and the deep ANNs can be trained robustly. Moreover, [24] demonstrated the implementation of self-recurrent connections in SNNs. It also showed that the self-recurrent structure can realize recurrent behaviors similar to the more complex RSNNs.

In our approach, the self-recurrent connection is only applied to the excitatory neuron. As expressed in (7), the neuron resets its membrane potential to 0 after firing. All the previous information accumulated in this neuron is lost. Despite the fact that the firing-and-resetting mechanism keeps the dynamics and oscillation of the network [5], there are still situations in which maintaining long-term memories of neurons is beneficial. Thus, self-recurrent connection plays the role of refreshing the memory. In this work, the self-recurrent connection is initialized to be a non-negative value then is trained by the learning algorithm. After training, the weights of self-recurrent connections are learned to determine how much previous information should be kept. Although the self-recurrent connections have other benefits as demonstrated in [24], [30], in this work, they are mainly used to mitigate the information loss caused by the firing-and-resetting mechanism. Thus, the temporal contextual information is refreshed and held in the internal states of the recurrent structure.

On the other hand, the excitatory neuron also accepts inhibition from the inhibitory neuron. Its membrane potential is depressed by a fixed amount when a spike comes from the inhibitory connection. In other words, while previous information is kept through the self-recurrent connection, the inhibitory connection determines when the existing information should be abandoned. Therefore, the inhibition serves as a gating mechanism to control the flow of information through neurons. In the meanwhile, the inhibitory neuron accepts inputs from presynaptic layer and the corresponding excitatory neuron. The weights connected to presynaptic layer are trained to learn when the inhibition should be generated. From the network perspective, the inhibitory connections also play roles such as filtering input signals, regulating network activities, and maintaining network dynamics [5], [31].

Moreover, the proposed LISR only contains recurrent connections inside itself. Thus, unlike most existing RSNN works that the networks only have one recurrent layer, this structured unit can be readily exploited as a basic building block for constructing multi-layered networks. Fig. 2 demonstrates a deep SNN implementing the proposed LISR. The network has $l+1$ hidden layers with full connections between the adjacent layers. Each layer is constructed by repeating LISRs. The LISRs of the same layer are independent without recurrent connections between each other. In addition, each LISR has three recurrent connections as introduced in this section. In Section IV, we demonstrate that this deep SNN structure can learn the spatial-temporal inputs information effectively.

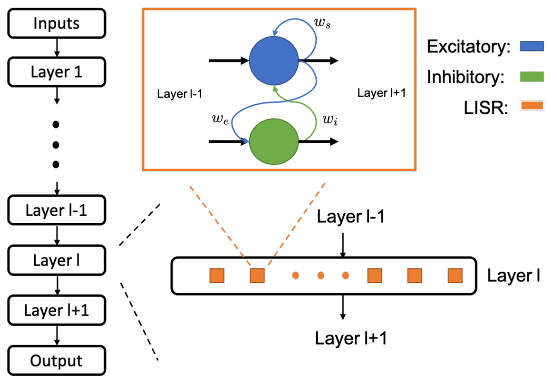

Fig. 2. Deep SNN based on LISR.

C. Backpropogation Method

Due to the structured connections of LISR, the BP method can be efficiently applied. In this work, we adopt the Temporal spike sequence learning via backpropagation (TSSL-BP) method proposed in [32] to train the whole network. TSSL-BP is a BP method directly training the SNNs. It captures the error backpropagation across two types of inter/intra-neuron dependencies and leads to state-of-the-art performance with low latency. The TSSL-BP is derived for feedforward SNNs and self-recurrent connections in [32] and [24]. In this paper, we follow the same idea but extend it to the proposed structure.

First, we denote the loss function as

$$L = \sum_{k=0}^{N_t} E[t_k], \tag{8}$$

where N_t is the total time steps and $E[t_k]$ the loss at t_k.

From (5), the membrane potential $u_p[t]$ of the neuron p at time t demonstrates contribution to all future fires and losses of the neuron. Therefore, the error gradient with respect to the presynaptic weight w_{pq} from neuron q to neuron p can be defined as

$$\begin{aligned}\frac{\partial L}{\partial w_{pq}} &= \sum_{k=0}^{N_t} \frac{\partial E[t_k]}{\partial w_{pq}} = \sum_{k=0}^{N_t}\sum_{m=0}^{k} \frac{\partial E[t_k]}{\partial u_p[t_m]}\frac{\partial u_p[t_m]}{\partial w_{pq}} \\ &= \sum_{m=0}^{N_t} a_q[t_m] \sum_{k=m}^{N_t} \frac{\partial E[t_k]}{\partial u_p[t_m]} = \sum_{m=0}^{N_t} a_q[t_m]\delta_p[t_m],\end{aligned} \tag{9}$$

where $\delta_p[t_m]$ denotes the error for neuron p at time t_m and is defined as:

$$\delta_p[t_m] = \sum_{k=m}^{N_t} \frac{\partial E[t_k]}{\partial u_p[t_m]} = \sum_{k=m}^{N_t} \frac{\partial E[t_k]}{\partial a_p[t_k]}\frac{\partial a_p[t_k]}{\partial u_p[t_m]}. \tag{10}$$

In this work, the neurons in the output layer are regular feedforward neurons without recurrent connection. Therefore, the weights of output neuron o are updated by

$$\frac{\partial L}{\partial w_{oq}} = \sum_{m=0}^{N_t} a_q[t_m] \sum_{k=m}^{N_t} \frac{\partial E[t_k]}{\partial a_o[t_k]}\frac{\partial a_o[t_k]}{\partial u_o[t_m]}, \tag{11}$$

where $\frac{\partial E[t_k]}{\partial a_o[t_k]}$ depends on the choice of the loss function.

For the neurons in the hidden layer, we derive the learning rule for the excitatory neuron e and the inhibitory neuron i in the layer l separately. In the rest of the paper, variables associated with neurons in the layer l have (l) as the superscript.

The weights update of neuron e and neuron i still follows the (9). However, due to their special recurrent connections, the derivations of $\delta_e^{(l)}$ and $\delta_i^{(l)}$, the error backpropagated from postsynaptic layer, are different.

For the excitatory neuron, in addition to the error signals from the postsynaptic layer, the error backpropagated from the self-recurrent connection and the excitatory recurrent connection should also be taken into consideration. Thus, the backpropagated error can be calculated by:

$$\begin{aligned}&\delta_e^{(l)}[t_m] = \\ &\sum_{j=m}^{N_t}\sum_{k=m}^{j} \frac{\partial a_e^{(l)}[t_k]}{\partial u_e^{(l)}[t_m]} \sum_{p=1}^{N^{(l+1)}} \left(\frac{\partial u_p^{(l+1)}[t_k]}{\partial a_e^{(l)}[t_k]} \frac{\partial E[t_j]}{\partial u_p^{(l+1)}[t_k]} \right) \\ &+ \sum_{j=m+1}^{N_t}\sum_{k=m}^{j} \frac{\partial a_e^{(l)}[t_k]}{\partial u_e^{(l)}[t_m]} \left(\frac{\partial u_e^{(l)}[t_k+1]}{\partial a_e^{(l)}[t_k]} \frac{\partial E[t_j]}{\partial u_e^{(l)}[t_k+1]} \right) \\ &+ \sum_{j=m+1}^{N_t}\sum_{k=m}^{j} \frac{\partial a_e^{(l)}[t_k]}{\partial u_e^{(l)}[t_m]} \left(\frac{\partial u_i^{(l)}[t_k+1]}{\partial a_e^{(l)}[t_k]} \frac{\partial E[t_j]}{\partial u_i^{(l)}[t_k+1]} \right).\end{aligned} \tag{12}$$

where $N^{(l+1)}$ denotes the number of neurons in the layer $l+1$.

By changing the order of summation, (12) is written as

$$\begin{aligned}\delta_e^{(l)}[t_m] &= \sum_{k=m}^{N_t} \frac{\partial a_e^{(l)}[t_k]}{\partial u_e^{(l)}[t_m]} \sum_{p=1}^{N^{(l+1)}} w_{pe}\delta_p^{(l+1)}[t_k] \\ &+ \sum_{k=m}^{N_t-1} \frac{\partial a_e^{(l)}[t_k]}{\partial u_e^{(l)}[t_m]} w_s \delta_e^{(l)}[t_k+1] \\ &+ \sum_{k=m}^{N_t-1} \frac{\partial a_e^{(l)}[t_k]}{\partial u_e^{(l)}[t_m]} w_e \delta_i^{(l)}[t_k+1],\end{aligned} \tag{13}$$

where $\delta_p^{(l+1)}[t_k]$ is the error of the neuron p in the layer $l+1$ at time t_k, $\delta_e^{(l)}[t_k+1]$ and $\delta_i^{(l)}[t_k+1]$ the errors of the excitatory neuron itself and the inhibitory neuron at time t_k+1 respectively.

(13) reveals that membrane potential $u_e^{(l)}$ of the excitatory neuron in layer l influences its (unweighted) PSC $a_e^{(l)}$ through spikes, and $a_e^{(l)}$ further affects the membrane potentials of its postsynaptic neurons. Its first term reflects that the error of the excitatory neuron is accumulated from the errors of all postsynaptic layer neurons. The second and third terms indicate that the errors are also backpropagated from the excitatory neuron itself and the inhibitory neuron.

Similarly, the error of inhibitory neuron can be obtained by

$$\begin{aligned}\delta_i^{(l)}[t_m] &= \sum_{k=m}^{N_t} \frac{\partial a_i^{(l)}[t_k]}{\partial u_i^{(l)}[t_m]} \sum_{p=1}^{N^{(l+1)}} w_{pi}\delta_p^{(l+1)}[t_k] \\ &+ \sum_{k=m}^{N_t-1} \frac{\partial a_i^{(l)}[t_k]}{\partial u_i^{(l)}[t_m]} w_i \delta_e^{(l)}[t_k+1],\end{aligned} \tag{14}$$

where the first term represents errors from postsynaptic layer and second term is the error from the excitatory neuron.

In addition, the calculation of the term $\frac{\partial a[t_k]}{\partial u[t_m]}$ is one of the key contributions of [32]. We do not repeat the steps but treat it as a known term in this paper.

III. Experimental Settings

In this work, the proposed LISR structure is evaluated on three datasets, speech dataset TI46 [25], neuromorphic speech dataset N-TIDIGITS [26], and neuromorphic image dataset DVS-Gesture [27]. The LISR networks in the experiments are formed such that all the hidden layers are composed of LISRs. In other words, a hidden layer with n neurons has $\frac{n}{2}$ LISRs with $\frac{n}{2}$ excitatory neurons and $\frac{n}{2}$ inhibitory neurons.

All reported experiments are conducted on an NVIDIA Titan XP GPU. The implementation of the proposed structure and the BP method is based on the Pytorch framework [33].

A. Parameter Settings

In the SNNs of the experiments, the fully connected weights between layers are initialized by the He Normal initialization proposed in [34]. The self recurrent weights are initialized to 0.5 while the weights of the excitatory recurrent connection and the inhibitory recurrent connection are fixed to 1 and −2, respectively. AdamW [35] is adopted as the optimizer.

The simulation step size is set to 1 ms. Other parameters are summarized in Table I. The hyperparameters and weights initialization are empirically tuned.

TABLE I
PARAMETER SETTINGS

Parameter	Value	Parameter	Value
τ_m	64 ms	τ_s	8 ms
V_{th}	1 mV	V_{rest}	0 mV
batch size	100	learning rate	0.0005

B. Loss Function

For the BP method used in this work, the loss function can be defined by any errors that measure the distance between the actual outputs and the desired outputs. In our experiments, since hundreds of time steps are required for simulating speech and video inputs, we choose the accumulated output PSCs to define the error which is similar to the firing count used in many existing works [36], [37].

We suppose the simulation time steps for a sample is N_t. Furthermore, for neuron o of the output layer, we define the desired output as d_o and real output as r_o where $r_o = \sum_{k=1}^{N_t} a_o[t_k]$ and d_o is manually determined. Therefore, the loss is determined by the square error of the outputs

$$L = \sum_{k=1}^{N_t} E[t_k] = \sum_{o}^{N^{(out)}} \frac{1}{2}(d_o - r_o)^2, \tag{15}$$

where $N^{(out)}$ is the number of neurons in the output layer.

Furthermore, the error at each time step is simply defined by the averaged loss through all the time steps:

$$E[t_k] = L/N_t, \quad E_o[t_k] = \frac{(d_o - r_o)^2}{2N_t}. \tag{16}$$

With the loss function defined above, the error δ can be calculated through the (10), (13), and (14).

C. Datasets

TI46 speech corpus [25] contains spoken English alphabets and digits audios from 16 speakers. In our experiments, the full alphabets subset of the TI46 is used. We name this subset TI46-Alpha in the rest of the paper. The TI46-Alpha has 4142 and 6628 spoken English examples in 26 classes for training and testing, respectively. The continuous temporal speech waveforms are preprocessed by Lyon's ear model [38] which is the same as the preprocessing steps in [22]. The sample rate of this dataset is 12.5 kHz. The decimation factor of Lyon's ear model is 125. The Matlab implementation of Lyon's ear model is available online [39]. Each sample is encoded into 78 channels. In our experiments, the preprocessed real-value intensities are directly applied as the inputs.

The N-TIDIGITS [26] is the neuromorphic version of the speech dataset Tidigits [40]. The original audios are processed by a 64-channel CochleaAMS1b sensor and recorded as the spike responses. The dataset contains both single-digit samples and connected-digit sequences with a vocabulary consisting of 11 digits including "oh," "zero" and the digits "1" to "9". In the experiments, only the single-digit samples are used. In total, there are 55 male and 56 female speakers with 2475 single-digit samples for training and the same number of samples for testing. In the original dataset, each sample has 64 input channels and takes about 0.9 s. To speed up the simulation, each sample is reduced to 300 time steps by compressing the time resolution from 1 us to 3 ms. During the compression, a channel has a spike at a certain time step in the preprocessed sample if it contains at least one spike in the corresponding time window of the original sample.

The DVS-Gesture dataset [27] consists of recordings of 29 different individuals (subjects) performing hand and arm gestures. The spikes are generated from natural motion. There are 122 trials in total. Each trial contains the recording for one subject by a dynamic vision sensor (DVS) camera under one of the three different lighting conditions. In each trial, 11 hand and arm gestures of the subject are recorded. We follow the same preprocessing procedures in [37]. Samples from the first 23 subjects are used for training and the other 6 subjects for testing. During preprocessing, the trials are separated into individual actions (gestures). The task is to classify the action sequence video into an action label. Each action (sample) lasts for about 6 s. In addition, two channels with 128×128 pixels in each channel are recorded. In the experiments, only the first 1.5 s of action video for each sample are used. We compress the temporal resolution to 5 ms which means it takes 300 time steps for each sample. Similar to the preprocessing of N-TIDIGITS, the input pixel has a spike at a certain time step in the preprocessed sample if it contains at least one spike in the corresponding 5 ms time window of the original sample.

IV. EXPERIMENTAL RESULTS

A. TI46-Alpha

Table II demonstrates the experimental results on TI46-Alpha dataset. The network for the proposed LISR structure only has one hidden layer with 800 neurons. In other words, the hidden layer contains 400 LISRs with 400 excitatory neurons and 400 inhibitory neurons. In the last two rows of Table II, we compare performance between the proposed structure and feedforward SNN. These two experiments have the same network size, learning rule, preprocessing steps, and hyperparameters. The only difference is the network structure in the hidden layer. As shown, by implementing the proposed structure, the network can achieve 5.03% performance improvement with almost the same number of tunable parameters. In addition, performance of the existing works on the TI46-Alpha dataset are also listed.

In [41], the LSM with randomly connected reservoir is adopted. The state vector of the reservoir is used to train the single readout layer using the non-spiking BP method. The result indicates that only training the single readout layer of a recurrent LSM with random recurrent connections is inadequate. [22] demonstrates performance of the deep RSNNs trained by a spike-train level BP method. Its experiment contains three hidden layers with two feedforward layers and only

TABLE II
ACCURACY ON TI46-ALPHA

Network Structure	Learning Rule	Hidden Layers	# Params	# Time Steps	# Epochs	Accuracy
LSM [41]	Non-spiking BP	2000	$52,000$	–	–	78%
Feedforward SNN [36]	HM2BP	800	$83,200$	700	200	89.92%
RSNN [22]	ST-RSBP	$400-400-400$	$363,313$	700	100	93.35%
Sr-SNN [24]	TSSL-BP	800	$84,000$	100	300	93.06%
Feedforward SNN	TSSL-BP	800	$83,200$	100	300	91.05%
LISR (This work)	TSSL-BP	800	$83,600$	100	300	**96.08%**

TABLE III
ACCURACY ON N-TIDIGITS

Network Structure	Learning Rule	Hidden Layers	# Params	# Time Steps	# Epochs	Accuracy
Feedforward SNN [36]	HM2BP	$250-250$	$81,250$	300	300	89.69%
GRU [26]	Non-spiking BP	$200-200-100$	$314,700$	–	–	90.90%
Phase LSTM [26]	Non-spiking BP	$250-250$	$818,750$	–	–	91.25%
Feedforward SNN	TSSL-BP	400	$30,000$	300	300	84.84%
LISR (This work)	TSSL-BP	400	$30,200$	300	300	**94.10%**
RSNN [22]	ST-RSBP	$400-400-400$	$351,241$	300	300	93.90%
ScSr-SNN [24]	TSSL-BP	$400-400-400$	$512,400$	300	300	95.07%
Feedforward SNN	TSSL-BP	$400-400-400$	$350,000$	300	300	91.03%
LISR (This work)	TSSL-BP	$400-400-400$	$350,600$	300	300	**96.65%**

TABLE IV
ACCURACY ON DVS-GESTURE

Network Structure	Learning Rule	Hidden Layers	# Params	# Time Steps	# Epochs	Accuracy
Feedforward SNN [42]	STBP	$P4-512$	$1,054,208$	60	100	87.50%
RNN [42]	Non-spiking BP	$P4-512$	$1,316,352$	60	100	52.78%
LSTM [42]	Non-spiking BP	$P4-512$	$5,250,560$	60	100	88.19%
Feedforward SNN	TSSL-BP	$P4-512$	$1,054,208$	300	100	88.19%
LISR (This work)	TSSL-BP	$P4-512$	$1,054,464$	300	100	**89.93%**

the second hidden layer has random recurrent connections. Moreover, [24] presents the network with the self-recurrent connections for each neuron in the hidden layer. The whole network is also trained by the TSSL-BP method.

Among all the results, our proposed method outperforms the result in [22] by 2.73%. Furthermore, it achieves the best performance with only a similar or smaller number of parameters which leads to high-efficient training and inference.

B. N-TIDIGITS

In Table III, we demonstrate performance of the proposed structure on the N-TIDIGITS dataset by two networks. The first network only has one hidden layer with 400 neurons while another network has three hidden layers with 400 neurons in each layer. All the hidden layers are implemented with the proposed LISR. Similarly, the feedforward SNNs with the same network size are also tested. By comparing with the feedforward counterparts, the proposed structure is not only effective for the single-layer network but also highly improves performance with multiple hidden layers. For the single-layer network, the proposed method can impressively improve performance by 9.26% compared to the feedforward SNN with a similar number of parameters.

Performance is also compared with the feedforward networks trained by HM2BP [36], RSNNs trained by ST-RSBP [22], and the Skip-Connected Self-Recurrent SNN (ScSr-SNN) trained by TSSL-BP [24]. As shown, the proposed method outperforms the state-of-the-art results in the existing works. In addition, the LISR structure also achieves more than 5% better performance than the widely-adopted recurrent structures of ANNs, the GRU and LSTM, on this dataset. One possible reason for the LISR network significantly outperforming ANNs is that the dataset is neuromorphic based with spikes as inputs. Thus, the SNNs are more likely to handle the spatial-temporal information inside the dataset effectively.

C. DVS-Gesture

In the experiments on the DVS-Gesture dataset, all the networks have the same size. The inputs are first processed by the pooling layer of 4×4 pooling kernel size. Thus, the inputs to the 512 neurons hidden layer have 2 channels with the size of 32×32 in each channel.

As demonstrated in Table IV, the proposed LISR structure can improve performance by up to 2.43% compared to the same size feedforward SNNs trained by TSSL-BP [32] or STBP [43]. In addition, performance of the proposed method also outperforms the ANN structures including vanilla RNN and LSTM. In [42], a rate-coding-inspired loss function is proposed for enhancing performance of ANNs on neuromorphic image datasets. However, our proposed method still achieves more than 1.74% performance improvement over the rate-coding-inspired loss function, and our method uses a much smaller number of parameters.

V. Discussion

From the results of the three datasets above, the proposed LISR structure presents powerful learning ability and demonstrates the best performance compared to the existing works. To better understand the proposed structure, in this section, we demonstrate more analysis of the effects of the LISR.

A. Firing Activity

To analyze firing events, we select two well-trained networks, one feedforward SNN and one LISR network. Both of the networks have only one hidden layer with 800 neurons. An alphabet sample with 'A' speech of the TI46-Alpha dataset is used for the experiment.

Fig. 3 presents the firing events of all neurons in the hidden layer of the two networks. Each blue dot in the figure represents a spike. For the LISR network, the first 400 neurons are excitatory and the other 400 neurons are inhibitory. As shown in Fig. 3, the activity of the LISR network is denser than the activity of the feedforward network. Even for the excitatory neurons which receive strong inhibition from the inhibitory neuron, more firing events are observed compared to the neurons in the feedforward network.

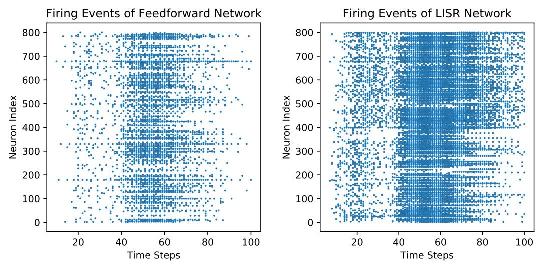

Fig. 3. Firing events on a TI46-Alpha sample.

Moreover, we calculate the firing rate of each neuron in the hidden layers of the feedforward network and the LISR network. In Fig. 4, the firing rates of the networks are categorized into seven groups. The number of neurons that have firing rates within certain thresholds is summed together. As shown in Fig. 4, about half of the neurons in the feedforward network are silent while only 14.375% neurons have firing rates greater than 10%. However, for the same input sample, only 23% neurons of the LISR network are silent. In the meanwhile, more than 48% neurons have firing rates greater than 20%, and 12.125% neurons have more than 30% firing rates.

As illustrated above, the LISR network enhances the network activity so that more neurons are involved in learning and information processing. Moreover, since the spatial-temporal information is passed through spikes, the neuron requires a certain level of activity to transfer and filter information. Furthermore, only when the activity is high enough can the neuron be sensitive to the small inputs. Therefore, the proposed structure benefits the network performance by refreshing and

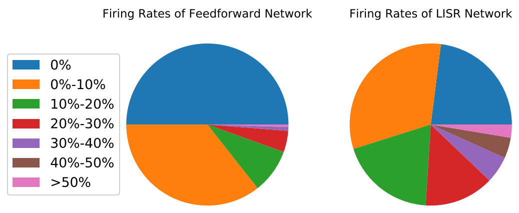

Fig. 4. Firing rates on a TI46-Alpha sample.

gating the information maintained inside the neurons as well as adjusting the network activities. On the other hand, although the overall firing rates of the LISR network are slightly higher, its firing activity is still sparse with more than half of neurons having a firing rate of 10% or lower. Thus, the sparsity of firing events, which is one of the crucial features of biologically inspired networks, is maintained.

B. Computational Complexity

The proposed method also contributes to the high-efficient training and inference of the network. For a hidden layer with n neurons, since the weights of excitatory and inhibitory connections are fixed and only self-recurrent connections are trained, the implementation of LISR only introduces $2.5n$ more parameters for the forward pass and $0.5n$ parameters for the backward pass. The number of additional recurrent weights is small compared to the parameters between layers. Since the computational cost of simulation is highly related to the number of parameters, the LISR network can enhance the network performance with almost no additional cost compared to feedforward SNN. In addition, the TSSL-BP [32], which can train networks over a short temporal window of a few time steps, is adopted as the learning rule for the proposed method to further reduce the computational overhead.

VI. Conclusion

In this paper, we proposed a novel recurrent structure called the Laterally-Inhibited Self-Recurrent Unit (LISR). We demonstrate the proposed structure and illustrate that the LISR can be easily applied to deep RSNNs. In addition, the learning rule is derived to be eligible to train this unique structure. In the results, the proposed method is evaluated on three datasets cover speeches, neuromorphic speeches, and neuromorphic images. The experimental results consistently show that the proposed structure can outperform the existing methods as well as the feedforward counterparts with a similar or smaller number of parameters. The impressive performance improvement comes from the refreshing and gating mechanism, and the regulation of network activities introduced by the LISR network. In the future, we would like to evaluate the effectiveness of the proposed structure on hardware. We hope this work would advance the research on structures of SNNs and the neuromorphic computing community.

References

[1] D. Fisher, I. Olasagasti, D. W. Tank, E. R. Aksay, and M. S. Goldman, "A modeling framework for deriving the structural and functional architecture of a short-term memory microcircuit," *Neuron*, vol. 79, no. 5, pp. 987–1000, 2013.

[2] K. Daie, M. S. Goldman, and E. R. Aksay, "Spatial patterns of persistent neural activity vary with the behavioral context of short-term memory," *Neuron*, vol. 85, no. 4, pp. 847–860, 2015.

[3] M. Oster, R. Douglas, and S.-C. Liu, "Computation with spikes in a winner-take-all network," *Neural computation*, vol. 21, no. 9, pp. 2437–2465, 2009.

[4] Y. Chen, "Mechanisms of winner-take-all and group selection in neuronal spiking networks," *Frontiers in computational neuroscience*, vol. 11, p. 20, 2017.

[5] G. Buzsaki, *Rhythms of the Brain*. Oxford University Press, 2006.

[6] K. Kar, J. Kubilius, K. Schmidt, E. B. Issa, and J. J. DiCarlo, "Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior," *Nature neuroscience*, vol. 22, no. 6, pp. 974–983, 2019.

[7] K. Rajaei, Y. Mohsenzadeh, R. Ebrahimpour, and S.-M. Khaligh-Razavi, "Beyond core object recognition: Recurrent processes account for object recognition under occlusion," *PLoS computational biology*, vol. 15, no. 5, p. e1007001, 2019.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] H. Jaeger, "The "echo state" approach to analysing and training recurrent neural networks-with an erratum note," *Bonn, Germany: German National Research Center for Information Technology GMD Technical Report*, vol. 148, no. 34, p. 13, 2001.

[10] A. Graves, A.-r. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," in *2013 IEEE international conference on acoustics, speech and signal processing*. IEEE, 2013, pp. 6645–6649.

[11] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *EMNLP*, 2014.

[12] A. Voelker, I. Kajić, and C. Eliasmith, "Legendre memory units: Continuous-time representation in recurrent neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 15 570–15 579.

[13] W. Maass, "Networks of spiking neurons: the third generation of neural network models," *Neural networks*, vol. 10, no. 9, pp. 1659–1671, 1997.

[14] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: A new framework for neural computation based on perturbations," *Neural computation*, vol. 14, no. 11, pp. 2531–2560, 2002.

[15] Y. Zhang, P. Li, Y. Jin, and Y. Choe, "A digital liquid state machine with biologically inspired learning and its application to speech recognition," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 11, pp. 2635–2649, 2015.

[16] Y. Jin and P. Li, "Ap-stdp: A novel self-organizing mechanism for efficient reservoir computing," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 1158–1165.

[17] G. Srinivasan, P. Panda, and K. Roy, "Spilinc: Spiking liquid-ensemble computing for unsupervised speech and image recognition," *Frontiers in neuroscience*, vol. 12, 2018.

[18] A. Morrison, M. Diesmann, and W. Gerstner, "Phenomenological models of synaptic plasticity based on spike timing," *Biological cybernetics*, vol. 98, no. 6, pp. 459–478, 2008.

[19] F. Ponulak and A. Kasiński, "Supervised learning in spiking neural networks with resume: sequence learning, classification, and spike shifting," *Neural computation*, vol. 22, no. 2, pp. 467–510, 2010.

[20] Q. Wang and P. Li, "D-lsm: Deep liquid state machine with unsupervised recurrent reservoir tuning," in *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE, 2016, pp. 2652–2657.

[21] G. Bellec, D. Salaj, A. Subramoney, R. Legenstein, and W. Maass, "Long short-term memory and learning-to-learn in networks of spiking neurons," in *Advances in Neural Information Processing Systems*, 2018, pp. 787–797.

[22] W. Zhang and P. Li, "Spike-train level backpropagation for training deep recurrent spiking neural networks," in *Advances in Neural Information Processing Systems*, 2019, pp. 7800–7811.

[23] A. Maes, M. Barahona, and C. Clopath, "Learning spatiotemporal signals using a recurrent spiking network that discretizes time," *PLoS computational biology*, vol. 16, no. 1, p. e1007606, 2020.

[24] W. Zhang and P. Li, "Skip-connected self-recurrent spiking neural networks with joint intrinsic parameter and synaptic weight training," *arXiv preprint arXiv:2010.12691*, 2020.

[25] M. Liberman, R. Amsler, K. Church, E. Fox, C. Hafner, J. Klavans, M. Marcus, B. Mercer, J. Pedersen, P. Roossin, D. Walker, S. Warwick, and A. Zampolli, "TI 46-word LDC93S9," Philadelphia, 1991. [Online]. Available: https://catalog.ldc.upenn.edu/docs/LDC93S9/ti46.readme.html

[26] J. Anumula, D. Neil, T. Delbruck, and S.-C. Liu, "Feature representations for neuromorphic audio spike streams," *Frontiers in neuroscience*, vol. 12, p. 23, 2018.

[27] A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza *et al.*, "A low power, fully event-based gesture recognition system," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 7243–7252.

[28] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge university press, 2002.

[29] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, and M. Ranzato, "Learning longer memory in recurrent neural networks." in *ICLR (Workshop)*, 2015.

[30] S. Li, W. Li, C. Cook, C. Zhu, and Y. Gao, "Independently recurrent neural network (indrnn): Building a longer and deeper rnn," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 5457–5466.

[31] M. Bartos, I. Vida, and P. Jonas, "Synaptic mechanisms of synchronized gamma oscillations in inhibitory interneuron networks," *Nature reviews neuroscience*, vol. 8, no. 1, pp. 45–56, 2007.

[32] W. Zhang and P. Li, "Temporal spike sequence learning via backpropagation for deep spiking neural networks," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[33] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035.

[34] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[35] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *International Conference on Learning Representations*, 2018.

[36] Y. Jin, W. Zhang, and P. Li, "Hybrid macro/micro level backpropagation for training deep spiking neural networks," *Advances in neural information processing systems*, vol. 31, pp. 7005–7015, 2018.

[37] S. B. Shrestha and G. Orchard, "Slayer: Spike layer error reassignment in time," *Advances in neural information processing systems*, vol. 31, pp. 1412–1421, 2018.

[38] R. Lyon, "A computational model of filtering, detection, and compression in the cochlea," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'82.*, vol. 7. IEEE, 1982, pp. 1282–1285.

[39] M. Slaney, "Auditory toolbox," *Interval Research Corporation, Tech. Rep*, vol. 10, no. 1998, p. 1194, 1998.

[40] R. G. Leonard and G. Doddington, "Tidigits speech corpus," *Texas Instruments, Inc*, 1993.

[41] P. Wijesinghe, G. Srinivasan, P. Panda, and K. Roy, "Analysis of liquid ensembles for enhancing the performance and accuracy of liquid state machines," *Frontiers in Neuroscience*, vol. 13, p. 504, 2019.

[42] W. He, Y. Wu, L. Deng, G. Li, H. Wang, Y. Tian, W. Ding, W. Wang, and Y. Xie, "Comparing snns and rnns on neuromorphic vision datasets: Similarities and differences," *Neural Networks*, vol. 132, pp. 108–120, 2020.

[43] Y. Wu, L. Deng, G. Li, J. Zhu, and L. Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in neuroscience*, 2018.