A Digital Liquid State Machine With Biologically Inspired Learning and Its Application to Speech Recognition

Yong Zhang, Peng Li, *Senior Member, IEEE*, Yingyezhe Jin, and Yoonsuck Choe, *Senior Member, IEEE*

***Abstract*— This paper presents a bioinspired digital liquid-state machine (LSM) for low-power very-large-scale-integration (VLSI)-based machine learning applications. To the best of the authors' knowledge, this is the first work that employs a bioinspired spike-based learning algorithm for the LSM. With the proposed online learning, the LSM extracts information from input patterns on the fly without needing intermediate data storage as required in offline learning methods such as ridge regression. The proposed learning rule is local such that each synaptic weight update is based only upon the firing activities of the corresponding presynaptic and postsynaptic neurons without incurring global communications across the neural network. Compared with the backpropagation-based learning, the locality of computation in the proposed approach lends itself to efficient parallel VLSI implementation. We use subsets of the TI46 speech corpus to benchmark the bioinspired digital LSM. To reduce the complexity of the spiking neural network model without performance degradation for speech recognition, we study the impacts of synaptic models on the fading memory of the reservoir and hence the network performance. Moreover, we examine the tradeoffs between synaptic weight resolution, reservoir size, and recognition performance and present techniques to further reduce the overhead of hardware implementation. Our simulation results show that in terms of isolated word recognition evaluated using the TI46 speech corpus, the proposed digital LSM rivals the state-of-the-art hidden Markov-model-based recognizer Sphinx-4 and outperforms all other reported recognizers including the ones that are based upon the LSM or neural networks.**

***Index Terms*— Hardware implementation, liquid-state machine (LSM), speech recognition, spike-based learning.**

I. Introduction

In-depth study on learning mechanisms of nervous systems [1]–[3] has motivated the development of spiking neural networks, which have been shown to be computationally more powerful than previous generations of neural networks based on McCulloch–Pitts neurons and threshold gates [4]. As a result, many recent research works have been geared toward more biologically inspired learning algorithms, network structures, and applications of spiking neural networks [5]–[11]. Bohte *et al.* [5] extended the widely used backward-propagation technique to spiking neural networks. Instead of training synaptic weights, Natschlager and Ruf [6] computed radial basis functions by training synaptic delays. To improve the stability of learning, Wade *et al.* [7] merged spike timing dependent plasticity with the Bienenstock–Cooper–Munro (BCM) model. Brader *et al.* [9] proposed a semisupervised learning model inspired by experimental observations for spiking neural networks. By introducing the mechanism of winner-take-all, McKinstry and Edelman [10] proposed a learning model to generate desired patterns of firing activities in the correct temporal order. Meng *et al.* [11] proposed the gene regulatory network (GRN)-BCM model for human behavior recognition, where plasticity parameters are regulated by a GRN.

On the other hand, as spiking neurons more closely resemble the behavior of neurons in nervous systems, they may also consume lower power than previous generations of neural networks when implemented in hardware [12]–[14]. In addition, it was shown that spiking neural networks are error resilient [15], a very appealing property for implementation in modern very-large-scale-integration (VLSI) technologies in which device reliability and process variation are the key challenges. In recent years, various neuromorphic chips were designed or fabricated to demonstrate their computational capability and low power consumptions for several applications [13]–[15].

In particular, inspired by the fact that neocortex processes a wide spectrum of information by stereotypical neural microcircuitry, the network model of the liquid-state machine (LSM), a particular form of spiking neural networks, was proposed and subsequently proven to be efficient for various tasks [8], [16]–[18]. Structurally, the LSM consists of a reservoir receiving input spike trains and a group of readout neurons receiving signals from the reservoir. With a group of neurons randomly connected by fixed synapses, the reservoir has a recurrent structure. This leads to decaying transient memories represented by the dynamic response of the reservoir to input spike trains. For this reason, the LSM is specially competent for processing temporal patterns such as speech signals [19], [20]. For readout neurons, Verstraeten *et al.* [19] used ridge regression to calculate

Manuscript received December 13, 2013; revised December 12, 2014; accepted December 27, 2014. Date of publication January 27, 2015; date of current version October 16, 2015.

Y. Zhang was with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA. He is now with Cadence Design Systems, Inc., San Jose, CA 95134 USA (e-mail: 86zhangyong@gmail.com).

P. Li and Y. Jin are with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: pli@tamu.edu; jyyz@tamu.edu).

Y. Choe is with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: choe@tamu.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2015.2388544

synaptic weights between the reservoir and readout neurons for classification tasks. Ghani *et al.* [20] trained output neurons using backpropagation-based multilayer perceptrons.

Compared with other standard methods for isolated utterance recognition, such as HMM-based [21], template-based [22], and acoustic-phonetic-based [23] approaches, the LSM is more biologically plausible and may be considered as a more general model for speech recognition. The internal LSM model parameters may be trained by extracting statistical information from the data without using acoustic and language models that are specific to targeted languages and datasets. Furthermore, the LSM is computationally powerful such that its performance on isolated word recognition can be comparable to state-of-art signal processing methods. Compared with other neural-network-based methods, such as Long Short-Term Memory nets [24] and multilayer perceptron-based classifiers [20], the LSM is either superior in performance or more hardware friendly and biologically plausible. However, the offline learning in [19] required large amounts of storage for intermediate results during training. Moreover, the learning rule used in backpropagation-based multilayer perceptrons [20] was arithmetically complicated and not local. These drawbacks complicate the network model and limit its potential application to real-time low-power hardware systems.

The key technical contribution of this paper is a new biologically motivated learning rule, which also lends itself to hardware-friendly implementation of the LSM. The proposed online learning allows the LSM to extract information from input patterns on the fly without needing intermediate data storage as required in offline learning methods. Our learning rule is local in the sense that each synaptic weight update is based only upon the firing activities of the corresponding presynaptic and postsynaptic neurons without incurring global communications across the neural network. Compared with the backpropagation-based learning, the locality of our approach is amenable to efficient parallel VLSI implementation. To reduce the complexity of the spiking neural network model without performance degradation for speech recognition, we study the impacts of synaptic models on the fading memory of the reservoir, which is strongly correlated with the computational power of the network. Moreover, all model parameters of the proposed LSM are digitized for digital CMOS implementation. We examine the tradeoffs between synaptic weight resolution, reservoir size, and recognition performance and present techniques to further reduce the hardware overhead targeting either a dedicated customized IC or an field-programmable gate array implementation [25]. Our simulation results show that in terms of isolated word recognition evaluated using the TI46 speech corpus, the proposed digital LSM rivals that of the state-of-the-art hidden Markov-model (HMM)-based recognizer Sphinx-4 [21], and outperforms all other reported speech recognizers including the ones that are based upon the LSM or neural networks.

The rest of this paper is organized as follows. Section II briefly introduces the LSM and its application to speech recognition. Section III presents the motivation behind the proposed learning rule, followed by the presentation of the design and implementation of the new rule in Section IV.

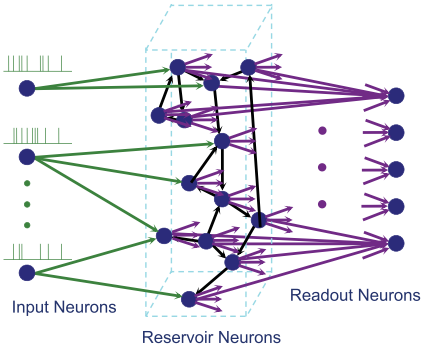


Fig. 1. Network structure of the LSM. Dots and arrows represent neurons and synapses, respectively. The neurons on the left provide input spike trains to the reservoir neurons in the middle. The reservoir receives input spike trains and projects through plastic synapses to the readout neurons on the right.

In Section V, the influence of synaptic models on fading memory is studied. Section VI presents design optimization techniques for efficient hardware-friendly implementation of the proposed LSM. Section VII evaluates the performance of the proposed LSM for speech recognition and the impacts of synaptic weight precision and reservoir size on recognition performance. The proposed LSM is also compared with several existing speech recognition techniques. Finally, this paper is concluded in Section VIII.

II. Liquid State Machine for Speech Recognition

A. General Network Structure

The network structure of the LSM is shown in Fig. 1. The reservoir in the middle is comprised of a set of neurons connected by fixed synapses generated either in a way approximating the spatial distribution of biological neurons [8] or purely randomly [26]. As multiple recurrent loops are created by these synaptic connections, the reservoir features transient behavior that memorizes information of its inputs in the past. Reservoir neurons and readout neurons are connected by plastic synapses whose weights are to be adjusted according to the adopted learning rule. Through its plastic synapses, each readout neuron receives a weighted sum of input signals from the reservoir neurons.

From Fig. 1, it is clear that the input signals are processed in two steps. The first step involves input neurons, reservoir neurons, and synapses connecting these neurons. Since the number of neurons in the reservoir is generally larger than that of the neurons providing inputs, in this step, the reservoir maps each input signal to its liquid response, a higher dimensional transient state. Note that this mapping is nonlinear in nature and that after being nonlinearly cast to a higher dimensional space, complex patterns are more likely to be linearly separable [27]. In the second step, the liquid response is projected to each readout neurons through plastic synapses

$$I_o(t) = \sum_i w_{i,o} \cdot r_i(t) \tag{1}$$

where t is time, $I_o(t)$ is the net input to a readout neuron, $r_i(t)$ is the output of the ith reservoir neuron, and $w_{i,o}$ is the weight of the synapse connecting the ith reservoir neuron and the readout neuron. Over the duration of $[0, T]$ of an input

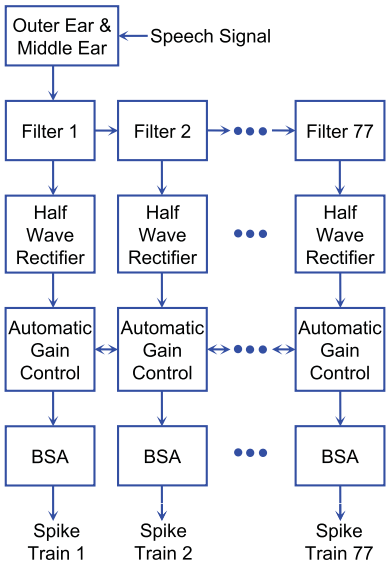


Fig. 2. Preprocessing of speech signals. The speech signal is processed by a filter representing the outer ear and middle ear followed by 77 cascaded bandpass filters modeling the cochlea. After each half-wave rectifier, the magnitude of the time-domain signal in each frequency band is compressed by an automatic control module. The resulting signal is converted to spike trains by the BSA.

temporal signal, the net integrated input to the readout neuron is

$$\int_0^T I_o(t) = \sum_i w_{i,o} \cdot \int_0^T r_i(t). \tag{2}$$

Since the net integrated input to each readout neuron is a linear combination of outputs of all reservoir neurons, each readout neuron can be viewed as a linear classifier with respect to the liquid responses. During the process of linear classification, only liquid responses produced by inputs of a certain class are expected to activate a specific readout neuron. In the corresponding feature space, a hyperplane defined by all $w_{i,o}$ separates these inputs from others. This linear classification problem was solved by determining the weights of the plastic synapses mathematically using the Ridge regression [19]. In [20], a hidden layer was added between the reservoir and readout neurons. Thus, backpropagation-based training algorithms for multilayer perceptrons were used such that the liquid responses of different classes do not have to be linearly separable.

B. Preprocessing of Speech Signals

To apply the LSM to speech recognition, speech signals shall be preprocessed and encoded by spike trains. A number of methods exist for this step, such as temporal based linear predictive coding [20] and techniques summarized in [19], which are the Hopfield coding [28], Mel-frequency cepstral coefficients (MFCCs) [29], [30], passive ear model [31], [32], and the Ben coding algorithm (BSA) [33]. For good performance and biological plausibility, we use the speech data preprocessed by the Lyon passive ear model [31], [34] and BSA.

The preprocessing stage combining the Lyon passive ear model and BSA is shown in Fig. 2. After the filter modeling the outer ear and middle ear, the speech signal is processed by the cochlea modeled by 77 cascaded bandpass filters with each extracting certain frequency band of the voice spectrum. Filter 1 extracts the highest frequency band and filter 77 the lowest. The signal extracted from each filter is further processed by a half wave rectifier and compressed by an automatic gain control module. The use of compression by AGC is inspired by the fact that the human ear can hear sound levels in a dynamic range of about 120 dB, while the firing frequency of neurons in response to sound only varies within about two orders of magnitude. After the compression, the time-domain signal is converted to a spike train by the BSA [33], where a stronger signal is converted to a spike train with a higher instantaneous spiking rate. In summary, the preprocessing stage converts the input speech signal into 77 parallel spike trains representing different frequency channels covering the entire voice spectrum.

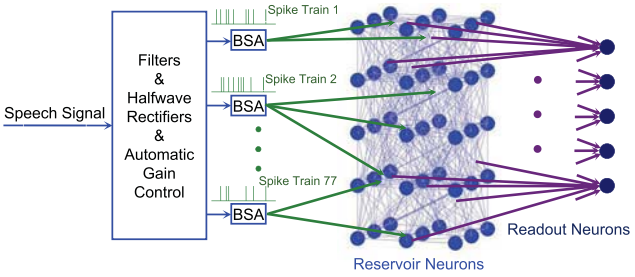


Fig. 3. Entire LSM system for speech recognition. Seventy-seven spike trains from the preprocessing stage are used as the inputs to the reservoir of the LSM. The BSA is implemented in each input neuron of the LSM.

C. Entire System for Speech Recognition

In the LSM of this paper, the reservoir has a regular grid structure [16]. It is an $k \times l \times m$ grid with an neuron at each grid point, as shown in Fig. 3. Synaptic connections are allocated randomly such that neurons that are closer to each other have a higher probability to be connected. The probability for forming a connection [8] is

$$P_{\text{connection}}(N_1, N_2) = k \cdot e^{-\frac{D^2(N_1,N_2)}{r^2}} \tag{3}$$

where N_1 and N_2 represent two neurons, $D(N_1, N_2)$ is the Euclidean distance between the two neurons, and k and r are two appropriately chosen constants.

The 77 spike trains produced by the preprocessing stage are fed to the LSM as the inputs. Conceptually, there is an input layer of 77 neurons whose outputs are generated by the BSA. The entire speech recognition system combining the preprocessing stage and the LSM is shown in Fig. 3.

III. Motivation for the Proposed Bio-Inspired Online Learning Rule

A. Hebbian Learning

As stated in the previous section, each readout neuron is viewed as a linear classifier. The online training process of a linear classifier is shown in Fig. 4 (left), where the hyperplane corresponding to the linear classifier is iteratively adjusted to

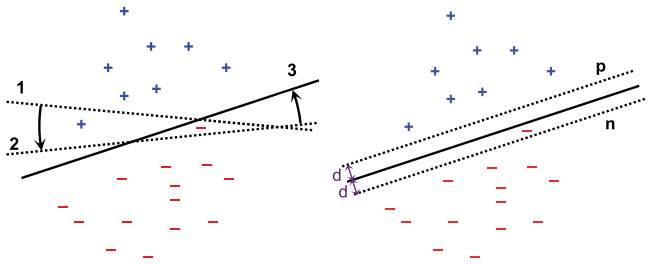


Fig. 4. Left: online training of a linear classifier. The orientation of the hyperplane is adjusted iteratively to separate two classes of data. Right: the linear classifier with a margin for improved generalization performance.

separate two classes of data in the feature space. Our goal is to find a spike-based learning rule for this training process.

In the neuroscience community, activity-based synaptic plasticity is believed to be the basic phenomenon in learning [1]. Based on this observation, the Hebb postulate was proposed. This widely accepted postulate states that neurons that fire together wire together [35]. More specifically, if the firing activity of neuron A tends to induce/inhibit spikes from another neuron B, the synaptic connection from neuron A to neuron B may be potentiated/depressed. The basic plasticity rule that follows the Hebb postulate is:

$$\tau_l \frac{dw}{dt} = u_{\text{pre}} u_{\text{post}} \tag{4}$$

where w is the synaptic weight, u_{pre} and u_{post} represent the activities of the presynaptic and postsynaptic neurons, respectively, and τ_l is the time constant corresponding to the learning speed. To include synaptic depression into the learning rule, different forms of covariance rules have been proposed, as summarized in [1].

B. Instability and Saturation

The Hebbian learning rules capture the most fundamental behavior of plastic synapses and are of great theoretical importance. However, there are practical instability issues associated with these rules, that is, synaptic weights may exhibit uncontrolled growth under the governing of these rules. One solution to this problem is to impose upper and lower limits to synaptic weights. However, with this solution, these rules suffer from a new problem of synaptic saturation in which all synaptic weights may be driven to the upper/lower limit such that the learning process is stopped and the network loses its capability to discriminate different input patterns.

To solve the synaptic saturation problem, several research works have been conducted. The BCM rule [36] uses a sliding threshold to modulate synaptic plasticity. Several techniques, e.g., the Oja rule [1], modify the Hebb rule to include various forms of synaptic normalization or synaptic competition. Theoretically, they all successfully solve the saturation problem, thus demonstrating their potential for practical applications. On the other hand, these techniques also have obvious drawbacks as a candidate for hardware implementation. First of all, the complicated computation involved in these learning rules requires a great deal of hardware resource. In addition, synaptic normalization/competition based rules are not local, that is, the learning dynamics of a synapse does not depend only on the firing activities of the presynaptic and postsynaptic neurons. The additional nonlocal communications between different parts of the neural network and the associated computations are not hardware friendly and may limit the scalability of these rules.

IV. Proposed Learning Rule

According to the discussions presented in the previous section, the main objective of our LSM-based online learning rule design is to develop a biologically inspired rule that is local and free from synaptic saturation. In addition, as a common issue in machine learning [37], overfitting shall be also considered in the design of learning algorithms. In this section, we first introduce an abstract learning rule that meets the above three requirements. A similar abstract rule for binary synapses has been used for two-layer feed-forward spiking networks in [9]. Then we specifically adopt and implement the abstract learning rule for the proposed LSM. Compared with the learning rule of [9], our implemented learning rule is greatly simplified and hence more hardware-friendly.

A. Abstract Learning Rule

We introduce the abstract local learning rule, which is local and free from saturation. The rule is then further improved in terms of its generalization performance.

As shown in Fig. 4 (left), the learning process is driven by incorrectly classified data points. In a neural network, if an inactive (active) neuron is desired to be active (inactive), the corresponding synapses that provide inputs to this neuron tend to be potentiated (depressed). Considering this for each output readout neuron in the LSM leads to

$$\begin{cases} \text{potentiation of } w_i, & \text{if } u_i = A \ \& \ u_r = I \ \& \ u_d = A \\ \text{depression of } w_i, & \text{if } u_i = A \ \& \ u_r = A \ \& \ u_d = I \end{cases} \tag{5}$$

where u_r and u_d represents the real (actual) and desired activities of the neuron, respectively, u_i is the activity of the ith presynaptic neuron, w_i is the corresponding synaptic weight, and A and I denote the active and inactive states, respectively. In a slightly different way, we use each presynaptic spike to trigger the update of the synaptic weight. Quantitatively, the value of w_i after a presynaptic spike may become

$$\begin{cases} w_i + \Delta w, & \text{if } u_r < u_\theta \ \& \ u_d > u_\theta \\ w_i - \Delta w, & \text{if } u_r > u_\theta \ \& \ u_d < u_\theta \end{cases} \tag{6}$$

where u_r and u_d are the quantitative values of neuron activity, u_θ is a threshold value used to determine whether the postsynaptic neuron is active or not, i.e., whether the data represented by the firing activities of all its presynaptic neurons is classified into one class or the other. It is clear that for any data point which is exactly on the hyperplane of the corresponding linear classifier, $u_d = u_\theta = \sum_i w_i \cdot u_i$.

For better generalization performance, a margin is introduced around the hyperplane for the linear classifier, as shown in Fig. 4 (right). By this way, the learning process is not only driven by incorrectly classified data but also by the

correctly classified data falling within the margin, i.e., between the two hyperplanes p and n. For any data on p (or n), $u_\theta + \Delta u = u_d = \sum_i w_i \cdot u_i$ (or $u_\theta - \Delta u = u_d = \sum_i w_i \cdot u_i$), where Δu corresponds to d in Fig. 4 (right).

Since a slow learning rate is usually used in neural networks for good performance [9], [38], correspondingly smaller learning steps are preferred in the training. To target at hardware implementation, we use discrete synaptic weights that have a finite resolution. To effectively reduce the learning rate, a stochastic weight update scheme is adopted. As such, the further modified learning rule states that the value of w_i following a spike from the ith presynaptic neuron may become:

$$\begin{cases} w_i + \Delta w \text{ with prob. } p_+, & \text{if } u_r < u_\theta + \Delta u \ \&\ u_d > u_\theta \\ w_i - \Delta w \text{ with prob. } p_-, & \text{if } u_r > u_\theta - \Delta u \ \&\ u_d < u_\theta \end{cases} \tag{7}$$

where Δw is the granularity of potentiation/depression, p_+ and p_- are the probabilities of potentiation and depression, respectively, and Δu is the margin.

The internal calcium concentration of a biological neuron is a good indicator of its instantaneous activity within a specified time window [9]. Replacing neuron activity u by calcium concentration c, (7) becomes

$$\begin{cases} w_i + \Delta w \text{ with prob. } p_+, & \text{if } c_r < c_\theta + \Delta c \ \&\ c_d > c_\theta \\ w_i - \Delta w \text{ with prob. } p_-, & \text{if } c_r > c_\theta - \Delta c \ \&\ c_d < c_\theta. \end{cases} \tag{8}$$

Note that this rule is mathematically similar to a more abstract rule for binary synapses in [9], which has been shown to be able to classify linearly separable patterns within a finite number of training iterations [39], [40].

B. Learning in Spiking Neural Networks

We describe the hardware-friendly implementation of the abstract learning rule described in the previous section. To mimic the behavior of a biological synapse, the implemented plasticity shall follow the Hebb's postulate that synaptic potentiation/depression takes place only if the presynaptic neuron tends to induce/inhibit firing of the postsynaptic neuron. From this perspective, the update of the synaptic weight after a presynaptic spike is expected to take the form

$$\begin{cases} w_i \to w_i + \Delta w \text{ with prob. } p_+, & \text{if } c_r > c'_\theta \\ w_i \to w_i - \Delta w \text{ with prob. } p_-, & \text{if } c_r < c'_\theta \end{cases} \tag{9}$$

where c'_θ is another threshold of calcium concentration used to determine if synaptic potentiation or depression is possible.

To take advantage of the merits of both forms of learning in (8) and (9), we combine these two principles. To be intuitive, we first illustrate these two rules in Fig. 5, where each subfigure is divided into several regions that show the learning activity of a synapse under different combinations of c_r and c_d. The subfigure on the left shows the abstract learning rule in (8), where the direction of learning is determined by the difference between the real activity of the postsynaptic neuron and its desired activity. The subfigure on the right shows (9), where the direction of the learning is determined by the activity of the postsynaptic neuron.

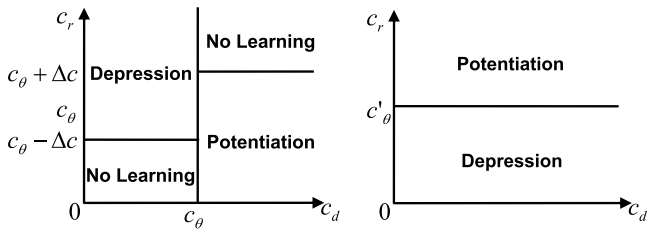


Fig. 5. Left: the abstract learning rule of (8) that determines potentiation or depression based upon the difference between the real and expected neuronal activities. Right: the Hebbian learning of (9) where a presynaptic spike leads to synaptic potentiation (depression) when the postsynaptic neuron is active (inactive).

We choose c'_θ to be the same as c_θ when combining these two principles. Since in a biological neuron synaptic learning is determined only by the actual activities of the presynaptic and postsynaptic neurons but not the desired activity, learning is expected to be independent of c_d, thereby only depending on c_r. To avoid synaptic saturation, we incorporate two stop learning regions corresponding to $c_r > c_\theta + \Delta c$ and $c_r < c_\theta - \Delta c$, respectively. In the region of $c_\theta - \Delta c < c_r < c_\theta + \Delta c$, the Hebbian learning is employed such that potentiation happens when $c_\theta < c_r < c_\theta + \Delta c$ and depression happens when $c_\theta - \Delta c < c_r < c_\theta$. Now, the combined learning rule dictates the learning of a synapse following a presynaptic spike according to:

$$\begin{cases} w_i \to w_i + \Delta w \text{ with prob. } p_+, & \text{if } c_\theta < c_r < c_\theta + \Delta c \\ w_i \to w_i - \Delta w \text{ with prob. } p_-, & \text{if } c_\theta - \Delta c < c_r < c_\theta. \end{cases} \tag{10}$$

Comparing (10) with (8) makes it clear that these two descriptions disagree in the region of $\{(c_d, c_r) | c_d < c_\theta, c_r > c_\theta\}$ and the region of $\{(c_d, c_r) | c_d > c_\theta, c_r < c_\theta\}$, where depression and potentiation occur, respectively, according to (8).

To this end, we employ a teacher signal to alter the actual (real) activity of the postsynaptic neuron to induce the desired learning of the synapse. Take region $\{(c_d, c_r) | c_d > c_\theta, c_r < c_\theta\}$ as an example. Since the desired learning according to (8) is potentiation, while depression occurs according to (10), the teacher signal is applied in such a way it moves each point in $\{(c_d, c_r) | c_d > c_\theta, c_r < c_\theta\}$ to $\{(c_d, c_r) | c_d > c_\theta, c_\theta < c_r < c_\theta + \Delta c\}$, where potentiation occurs according to (10). The effects of the teacher signal under various scenarios are shown by the arrowed lines in Fig. 6.

More specifically, the application of the teacher signal modulates the activity of the postsynaptic neuron such that its calcium concentration is driven to

$$\begin{cases} [c_\theta, c_\theta + \Delta c], & \text{if } c'_r < c_\theta \ \&\ c_d > c_\theta \\ [c_\theta - \Delta c, c_\theta], & \text{if } c'_r > c_\theta \ \&\ c_d < c_\theta \end{cases} \tag{11}$$

where c'_r is the internal calcium concentration of the postsynaptic neuron without applying the teacher signal.

The teacher signal of a postsynaptic neuron is implemented as a large constant current. In the case of a positive teacher signal, which increases the activity of the corresponding

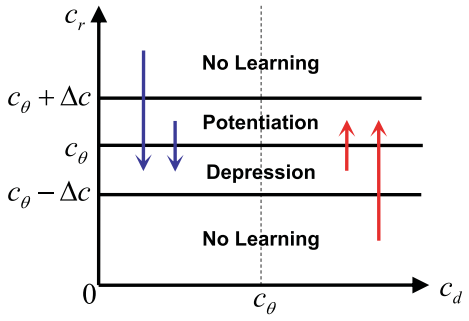


Fig. 6. Proposed learning rule. The eight regions shows how different combinations of c_d and c_r of the postsynaptic neuron determine the plasticity of a synapse. The four arrows show how the teacher signal drives the output neuron activity to the desired region.

postsynaptic neuron, a large constant current is injected to the targeted neuron if $c_d > c_\theta$ and $c_r < c_\theta + \delta$, where δ is a value in $[0, \Delta c]$, such that the teacher signal moves the current state of (c_d, c_r) deep into the region of $\{(c_d, c_r)|c_d > c_\theta, c_\theta < c_r < c_\theta + \Delta c\}$, where potentiation is induced according to (10). Note that the teacher signal is stopped when $c_r > c_\theta + \delta$. A negative teacher signal works in a complimentary way. The quantitative model of the teacher signal is introduced in the next section.

The use of the teacher signal in our proposed learning rule implements a supervised learning scheme for the LSM. We label the readout neurons by their corresponding class label R_i $(i = 1, 2, 3, \ldots)$. The goal of training is that when an input signal in class i is applied to the system shown in Fig. 3, R_i is the most active readout neuron, i.e., R_i emits more spikes than all other readout neurons. Thus, during training, a positive teacher signal is applied to R_i, while negative teacher signals are applied to other readout neurons.

To summarize, the proposed learning rule described by (10) and (11) is free from saturation, Hebbian and local, i.e., synaptic plasticity operates only upon the activities of the presynaptic and postsynaptic neurons. In addition, the performance of generation is ensured using margin Δc.

C. Models for Implementing the Proposed Learning Rule

In the LSM, we adopt the widely used leaky integrate-and-fire (LIF) model for each reservoir and readout neuron. The dynamics of the membrane potential of a neuron is described by the following differential equation:

$$\frac{dv_m}{dt} = -\frac{v_m}{\tau_m} + \sum_i \sum_j w_{mi} \cdot s(t - t_{ij} - d_i) \tag{12}$$

where v_m and τ are the membrane potential of the mth neuron and the time constant of its first-order dynamics, respectively, i and j are the indices of the presynaptic neurons and spikes from them, respectively, w_{mi} is the weight of the synapse connecting the ith presynaptic neuron to the mth neuron, t_{ij} is the spiking time of the jth spike emitted from the ith presynaptic neuron, d_i is the corresponding synaptic delay, and $s(\cdot)$ is the dynamic response of a synapse to an input spike. Including the teacher signal to the neuron model gives

$$\frac{dv_m}{dt} = -\frac{v_m}{\tau_m} + \sum_i \sum_j w_{mi} \cdot s(t - t_{ij} - d_i) + i_t(c) \tag{13}$$

where $i_t(\cdot)$ is the injected current due to the teacher signal, modeled as a function of calcium concentration c.

The above mode is digitized for hardware implementation as follows:

$$V_m^n = V_m^{n-1} - \frac{V_m^{n-1}}{\tau_m} + \sum_i \sum_j W_{mi} \cdot S(T^n, T_{i,j} + D_i) + I_t^{n-1} \tag{14}$$

where capitalized letters represent the digitized variables, and the superscripts of V, T, and I_t are the indices of time steps. If the membrane voltage of a neuron reaches or exceeds V_{th}, it fires and then resets its membrane voltage to resting potential V_{rest}. Following each fired spike, there is an absolute refractory period τ_{refrac} within which the neuron cannot fire again.

The dynamics of calcium concentration c is

$$\frac{dc}{dt} = -\frac{c}{\tau_c} + \sum_i \delta(t - t_i) \tag{15}$$

where τ_c is the time constant for the first-order dynamics of calcium concentration c and i is the index of spikes emitted from the neuron itself. From this differential equation, it is clear that the internal calcium concentration level of the neuron increases with its firing frequency. The digitized dynamics is

$$C^n = C^{n-1} - \frac{C^{n-1}}{\tau_c} + \sum_i \delta_{T^n, T_i} \tag{16}$$

where the same conventions for capitalization and superscripts as mentioned above are used, $\delta_{i,j}$ is the Kronecker delta whose value is 1 if $i = j$, and 0 if $i \neq j$.

Let $W_{\max}$ and $W_{\min}$ denote the upper and lower limits of weight values. To digitize synaptic weights, we also denote the quantization step size $W_{\max} - W_{\min}/2^b - 1$ by ΔW, where b is the number of bits used to represent each weight. In the digitized learning rule, the update of synaptic weights is initialized by a spike emitted by the presynaptic neuron. If a presynaptic fires at time T_n, then synaptic potentiation, i.e., $W^n = W^{n-1} + \Delta W$, happens with a probability of p_+ if the following conditions are satisfied:

$$\begin{cases} C_\theta < C^{n-1} < C_\theta + \Delta C \\ W^{n-1} < W_{\max} \end{cases} \tag{17}$$

where C_θ is the threshold of calcium concentration. Similarly, synaptic depression, i.e., $W^n = W^{n-1} - \Delta W$, happens with a probability of p_- when the following requirements are met:

$$\begin{cases} C_\theta - \Delta C < C^{n-1} < C_\theta \\ W^{n-1} > W_{\min}. \end{cases} \tag{18}$$

D. Comparison With Other Learning Rules

In recent years, several biologically inspired learning rules that can be potentially used for the LSM have been proposed. By training the output neurons to fire or not to fire by decoding the information in input spiking patterns, the so-called tempotron neuron [41] was designed for binary classification of spatiotemporal patterns. However, the learning rule used was offline and required a large amount of

memory. Thus, it has limited application to hardware based systems. More recently, several techniques, namely, remote supervised method (ReSuMe) [42], chronotron [43], and Spike Pattern Association Neuron (SPAN) [44], were proposed for processing temporally coded information by online learning. With these biologically inspired learning algorithms, neural networks were trained to reproduce spike trains with precise spike timing. As a spiking analogy of the Widrow–Hoff rule [45], ReSuMe minimized the error between the output and target spike trains during the training. The rule was also local and hence amenable to hardware implementation. The work of chronotron [43] proposed two learning rules: E-learning and I-learning. Similar to ReSuMe, E-learning minimized the error function measured by a new distance metric for improved continuity. Since E-learning was offline, a similar but more biologically plausible rule, I-learning, was proposed for online learning. In SPAN [44], to simplify the error calculation, spiking signals were transformed to analog signals, which, however, is not well suited for large-scale VLSI realization using digital CMOS circuits.

These learning rules demonstrated good performance on reproducing spike trains with precise timing information, while the proposed rule in this paper is specifically optimized for challenging real-life classification tasks. As shown in following sections, the proposed learning rule consistently produces high classification rates on isolated word speech recognition with various settings, while tests on real-world classification tasks were not reported for tempotron [41], ReSuMe [42], and chronotron [43]. In addition, tempotron was only designed for binary classification tasks. Last but not least, because of its hardware friendliness, the proposed learning rule is well suited for VLSI implementation.

V. Influence of Synaptic Models on LSM Performance and Implications on Hardware Implementation

Theoretical studies show that the computational performance of a recurrent neural network depends critically on its dynamics [46]–[48]. For superior computational performance, the reservoir in a LSM shall operate at the edge of chaos, where the system has long lasting transients (fading memory). This is in sharp contrast to an ordered or chaotic system, where the state evolves to a simple steady state or stable limit cycle (a state cycle relatively short in time) in the former, while exhibits chaotic behavior in the latter [47]. This implies that maintaining a sufficiently long fading memory is of great importance in boosting the computational performance of the LSM in terms of its separation property [8], [47]. Since fading memory arises from recurrent loops formed by synapses, the model of synapse may have a great impact on the length of fading memory, and consequently the LSM performance. However, this issue has not been well studied. In addition, for hardware-based LSMs, evaluating the complexity of different synaptic models and their impacts on the LSM performance at the same time provides design guidance to minimize hardware cost. We empirically study the influence of synaptic models with different complexities on the length of fading memory and the performance of the LSM.

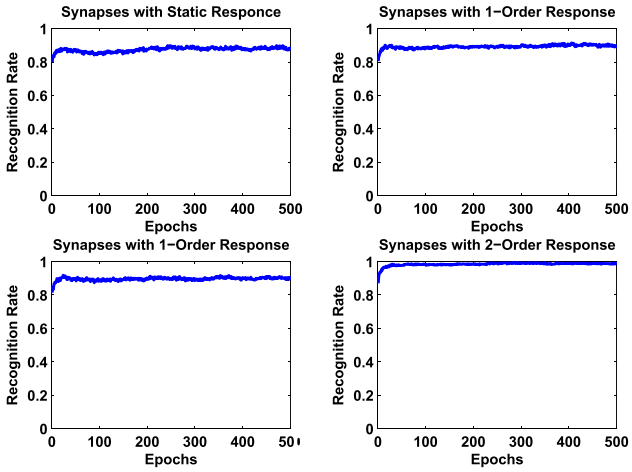


Fig. 7. Top left: performance of the LSM using synapses with static response. Top right: performance of the LSM using the first-order synaptic model with a time constant of 4 ms. Bottom left: performance of the LSM using the first-order synaptic model with a time constant of 8 ms. Bottom right: performance of the LSM using the second-order synaptic model.

A. Impacts of Synaptic Models on LSM Performance

Since our LSM is hardware oriented, simple synaptic models are preferred over complicated ones. Therefore, we first test the performance of the LSM with the simplest static synaptic model, i.e., the model's synaptic response to an incoming spike is static, or in other words, the model's synaptic response function is the Dirac delta function $\delta(\cdot)$, as used in many research works [13]–[15]. With this choice, the overall LIF model becomes

$$\frac{dv_m}{dt} = -\frac{v_m}{\tau_m} + \sum_i \sum_j w_{mi} \cdot \delta(t - t_{ij} - d_{ij}). \tag{19}$$

We test this model by training the LSM with the same settings used in [19] except that the proposed spike-based learning rule is adopted. The more detailed information of the LSM including settings and parameter values are described in Section VII. To study the influence of synaptic models on network performance, we use 500 utterances of digits 0–9 with five-fold cross validation to test the LSM. We train the LSM for 500 iterations and test its classification rate on the fly. The results are plotted in the top left panel of Fig. 7. It is observed that the recognition rate reaches about 88.85% with a standard deviation of 0.44%. Note that the finally reached performance is reported as the averaged classification rate of the last 20 epochs, i.e., epochs from 481 to 500. The same evaluation method is used hereinafter.

For comparison, we test the LSM performance with dynamical behavior involved in the synaptic model, where the synapse has a first-order dynamical response to a presynaptic spike. The corresponding LIF model is described as

$$\frac{dv_m}{dt} = -\frac{v_m}{\tau_m} + \sum_{i,j} w_{mi} \cdot \frac{1}{\tau^s} e^{-\frac{t-t_{ij}-d_{ij}}{\tau^s}} \cdot H(t - t_{ij} - d_{ij}) \tag{20}$$

where $\tau^s = 4$ ms is the time constant of the first-order response used for all synapses, $H(\cdot)$ is the Heaviside

step function, and $1/\tau^s$ normalizes the first-order response function. Without modifying any other models or parameters, solely adding the first-order dynamical response in the synaptic model improves the recognition performance to 90.49% with a standard deviation of 0.38%, as shown in Fig. 7 (top right). We also test the LSM performance using a different time constant of 8 ms. As shown in Fig. 7 (bottom left), the recognition rate is increased slightly to 90.73% with a standard deviation of 0.38% accordingly.

We further test the network performance with a second-order dynamic synaptic model

$$\frac{dv_m}{dt} = -\frac{v_m}{\tau_m} + \sum_i \sum_j \frac{w_{mi} \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_1^s}} \cdot H(t-t_{ij}-d_{ij})}{\tau_1^s - \tau_2^s} - \sum_i \sum_j \frac{w_{mi} \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_2^s}} \cdot H(t-t_{ij}-d_{ij})}{\tau_1^s - \tau_2^s} \quad (21)$$

where τ_1^s and τ_2^s are the time constants of the second order response, and the term $1/\tau_1^s - \tau_2^s$ normalizes the second-order dynamical response function. For excitatory synapses, τ_1^s and τ_2^s are set to 4 and 8 ms, respectively. For inhibitory synapses, τ_1^s and τ_2^s are set to 4 and 2 ms, respectively. As shown in Fig. 7 (bottom right), the recognition rate is further improved to 99.09% with a standard deviation of 0.04%.

The above results show that the use of synaptic models with higher orders dynamics significantly improves the LSM performance. A more complicated synaptic model with short-term plasticity (STP) was used in [19]. We compare the LSM performance reported in [19] with that of our LSM with a simpler second-order synaptic model in Section VII-F. This comparison shows that our LSM achieves a comparable performance. Avoiding incorporating STP in the synaptic model is very beneficial for hardware-friendly implementation since the state variables and computations associated with STP can be eliminated, leading to a reduced hardware complexity.

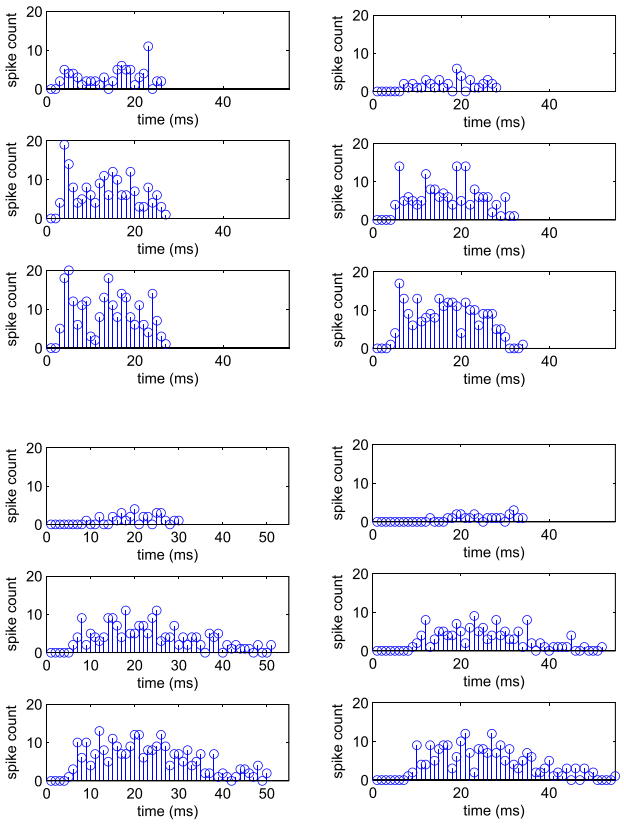


Fig. 8. Top left: the responses of the reservoir using the static synaptic model. With the input spike trains ending at 23 ms, the reservoir shows little temporal memory. With an ascending firing frequency for the input spike trains, the top, middle, and bottom plots show that only the magnitude of the reservoir response increases with the input firing frequency. Top right: the reservoir responses with the first-order synaptic model whose time constant is 4 ms. Different from the previous case, the reservoir exhibits temporal memory whose duration increases with the firing frequency of the input spike trains. Bottom left: the reservoir responses with the first-order synaptic model whose time constant is 8 ms. Compared with the case of the first-order model with a time constant of 4 ms, temporal memory is extended in this case. Bottom right: the reservoir responses with the second-order synaptic model. The temporal memory here is longer than those of the first-order models.

B. Fading Memory—Linking Synaptic Model to LSM Performance

It is evident from the discussions of the previous section that the LSM performance can be significantly improved using a dynamic synaptic model especially when a second-order synaptic model is used. To shed light on this key observation, we examine the impacts of the synaptic model on the fading memory of the reservoir as fading memory plays a critical role in the temporal pattern memorizing capability of a recurrent network.

The simulated impacts of synaptic models on the fading memory of the reservoir are shown in Figs. 8 and 9. In each panel of Fig. 8, the three plots show the activity of the reservoir in response to input spike trains ending at 23 ms. Each plot shows the total number of spikes in the reservoir within each 1 ms bin. The top plot of each panel uses the same input spike trains. The frequency of the input spike trains are 3 times and 10 times higher for the plots in the middle and bottom of all panels, respectively. As shown in Fig. 8 (top left), the responses of the reservoir vanish shortly after the end of the input spike trains with only the amplitude of the reservoir firing activity increasing with the input frequency, indicating little temporal memory of the reservoir. Fig. 8 (top right) shows that the use of a first-order synaptic model with a time constant of 4 ms makes the reservoir exhibit temporal memory, that is, the reservoir activity persists for a period of time after the end of the input spike trains. Increasing the time constant of the first-order synaptic model extends the temporal memory, as shown in Fig. 8 (bottom left). In addition, the network performance also increases accordingly. It shall be noted that increasing the synaptic model time constant only dramatically increase the fading memory when strong (high-frequency) input spikes are applied to the LSM, as shown in the bottom two plots of the panel. As shown in Fig. 8 (bottom right), the second-order synaptic model further extends the fading memory for the three input cases. Fig. 9 shows the averaged reservoir fading memory over all 500 isolated word utterances from the TI46 speech corpus used as the inputs to the LSM. Fading memory is recorded from the end of each utterance.

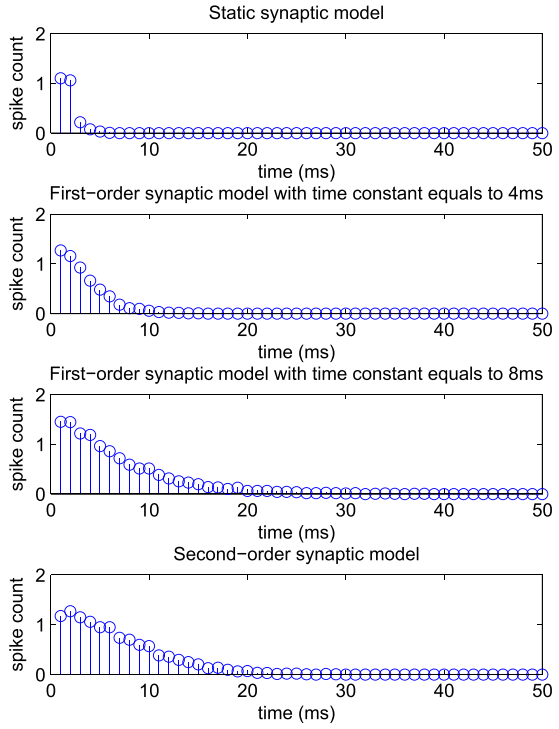


Fig. 9. Fading memory of the reservoir with different synaptic models averaged over 500 utterances. Fading memory is recorded starting from the end of each utterance.

In other words, the reservoir response to an input is recorded starting from the end of the input signal. The durations of the averaged memories in Fig. 9 are consistent with the results shown in Fig. 8.

These two figures show that higher order synaptic dynamics helps the reservoir produce longer fading memory, which further leads to richer dynamics and improved recognition performance. With higher order synaptic dynamics, the influence of an input spike to the postsynaptic neuron is spread out in time. The resulted long lasting effects promote the interactions between asynchronously emitted spikes throughout the recurrent reservoir. These interactions create rich dynamics that forms short-term memory and enhance the readout neurons' ability to recognize the given input patterns.

VI. Hardware-Friendly Design Optimization

A. Simplified Division Operations

To implement the proposed LSM in hardware, several key modules including the neuron model and the synaptic model of (16) and (21), and the learning rule of (17) and (18) must be efficiently realized. Several important arithmetic operations are involved in the above modules. Among these, addition and comparison operations [51] and random number generation [52] can be efficiently realized in VLSI. However, division operations are generally expensive to implement in hardware.

In (16) and (21), division operations are used to calculate the spontaneous exponential decrease of each neuron's calcium concentration and membrane potential. Lookup tables (LUTs) have been used to implement the exponential decrease of digitized variables [15]. However, one limitation of this approach is that its precision is limited by the affordable LUT area overhead.

It is useful to note that the time constants of calcium concentration and membrane voltage of our LSM are fixed across the entire neural network. Hence, a general-purpose division module is not necessary. The division of a binary number n by another number m can be easily realized when $m = 2^k$ for some nonnegative integer k. In this case, division can be done by simply right shifting the binary number n by k bits. Shifters are much cheaper to implement [51]. To explore this, we set the time constants of calcium concentration and membrane voltage to some powers of two. This choice does not have any significant negative impact on the LSM performance according to our experimental study.

B. Optimization of Synaptic Models

Efficient realization of synaptic models is critical as the number of synapses is usually much greater than that of neurons in a neural network. Consider the LSM for speech recognition as an example. To fully connect hundreds of reservoir neurons and 10 output neurons already requires thousands of plastic synapses, not to mention the synapses between the reservoir neurons.

To reduce the implementation complexity of synaptic models, we first take a closer look at these models. We rewrite (21), which describes the dynamics of both the neuron and its incident synapses

$$\frac{dv_m}{dt} = -\frac{v_m}{\tau_m} + \sum_i I_i \tag{22}$$

$$I_i = \sum_j w_{mi} \cdot \frac{1}{\tau_1^s - \tau_2^s} e^{-\frac{t-t_{ij}-d_{ij}}{\tau_1^s}} \cdot H(t - t_{ij} - d_{ij}) - \sum_j w_{mi} \cdot \frac{1}{\tau_1^s - \tau_2^s} e^{-\frac{t-t_{ij}-d_{ij}}{\tau_2^s}} \cdot H(t - t_{ij} - d_{ij}) \tag{23}$$

where (22) and (23) are the models of membrane voltage and the ith incident synapse, respectively, and I_i is the current injected to the neuron from the ith incident synapse. Without loss of generality, let the values of all t_{ij}s be positive. Interestingly, the expressions of $e^{-t-t_{ij}-d_{ij}/\tau_1^s} \cdot H(t - t_{ij} - d_{ij})$ and $e^{-t-t_{ij}-d_{ij}/\tau_2^s} \cdot H(t - t_{ij} - d_{ij})$ in (23) are, respectively, the solutions to the following differential equations:

$$\begin{cases} \dfrac{dx}{dt} = -\dfrac{1}{\tau_1} x + \delta(t - t_{ij} - d_{ij}) \\ x(t)|_{t=0} = 0 \end{cases} \tag{24}$$

and

$$\begin{cases} \dfrac{dx}{dt} = -\dfrac{1}{\tau_2} x + \delta(t - t_{ij} - d_{ij}) \\ x(t)|_{t=0} = 0. \end{cases} \tag{25}$$

By the principle of superposition, it is not difficult to see that the expressions of $w_{mi} \sum_j \cdot e^{-t-t_{ij}-d_{ij}/\tau_1^s} \cdot H(t - t_{ij} - d_{ij})$ and $w_{mi} \sum_j \cdot e^{-t-t_{ij}-d_{ij}/\tau_2^s} \cdot H(t - t_{ij} - d_{ij})$ are, respectively, the solutions to

$$\begin{cases} \dfrac{dx}{dt} = -\dfrac{1}{\tau_1} x + w_{mi} \sum_j \cdot \delta(t - t_{ij} - d_{ij}) \\ x(t)|_{t=0} = 0 \end{cases} \tag{26}$$

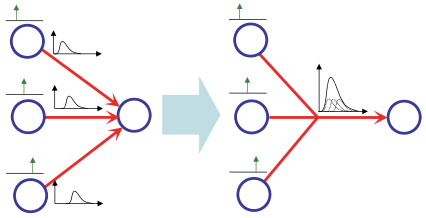


Fig. 10. Merging of linear synapses. The three synapses incident to a neuron are effectively merged into a single synapse.

and

$$\begin{cases} \dfrac{dx}{dt} = -\dfrac{1}{\tau_2}x + w_{mi}\sum_j \cdot \delta(t - t_{ij} - d_{ij}) \\ x(t)|_{t=0} = 0. \end{cases} \tag{27}$$

From a hardware-design point of view, (26) and (27) immediately suggest that a second-order synaptic model can be modeled using only two state variables associated with these two differential equations. This fact can be leveraged in design for efficient implementation of synaptic models.

To further minimize the implementation cost of a large number of synapses, we explore the linearity of the synaptic models to reduce the effective number of synapses that must be realized [53], [54], as shown in Fig. 10. We first reexamine the summed synaptic current of (23)

$$\begin{aligned} \sum_i I_i &= \sum_{i,j} w_{mi} \frac{e^{-\frac{t-t_{ij}-d_{ij}}{\tau_1^s}} - e^{-\frac{t-t_{ij}-d_{ij}}{\tau_2^s}}}{\tau_1^s - \tau_2^s} H(t - t_{ij} - d_{ij}) \\ &= \sum_{i,j} \frac{w_{mi}}{\tau_1^s - \tau_2^s} \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_1^s}} \cdot H(t - t_{ij} - d_{ij}) \\ &\quad - \sum_{i,j} \frac{w_{mi}}{\tau_1^s - \tau_2^s} \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_2^s}} \cdot H(t - t_{ij} - d_{ij}). \end{aligned} \tag{28}$$

By the principle of superposition, the following two terms from the right-hand side of (23):

$$\begin{aligned} &\sum_i w_{mi} \sum_j \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_1^s}} \cdot H(t - t_{ij} - d_{ij}) \\ &\qquad = \sum_{i,j} w_{mi} \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_1^s}} \cdot H(t - t_{ij} - d_{ij}) \end{aligned} \tag{29}$$

and

$$\begin{aligned} &\sum_i w_{mi} \sum_j \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_2^s}} \cdot H(t - t_{ij} - d_{ij}) \\ &\qquad = \sum_{i,j} w_{mi} \cdot e^{-\frac{t-t_{ij}-d_{ij}}{\tau_2^s}} \cdot H(t - t_{ij} - d_{ij}) \end{aligned} \tag{30}$$

are, respectively, the solutions to the same two linear differential equations with multiple inputs shown by

$$\begin{cases} \dfrac{dx}{dt} = -\dfrac{1}{\tau_1}x + \sum_i w_{mi}\sum_j \cdot \delta(t - t_{ij} - d_{ij}) \\ x(t)|_{t=0} = 0 \end{cases} \tag{31}$$

and

$$\begin{cases} \dfrac{dx}{dt} = -\dfrac{1}{\tau_2}x + \sum_i w_{mi}\sum_j \cdot \delta(t - t_{ij} - d_{ij}) \\ x(t)|_{t=0} = 0. \end{cases} \tag{32}$$

For example, (31) represents the same linear differential equation of (26) but with more inputs. Therefore, the solution to (31) is the linear combination of solutions to (26) with all inputs applied separately. The same principle applies to (27) and (32). This implies that the summed synaptic current of (23) or (28) can be efficiently evaluated by solving the two state variables in (31) and (32). As in (28), these two state variables shall be divided by $\tau_1^s - \tau_2^s$. For this, we use the shifter based division introduced in the previous section by setting τ_1^s, τ_2^s, and $\tau_1^s - \tau_2^s$ to some powers of two.

The above analysis can lead to a significant reduction of hardware implementation cost of linear synaptic models we consider in this paper. It is clear that instead of having two state variables for each second-order synapse, the state variables of all synapses incident to the same postsynaptic neuron can be effectively merged into only two first-order synapses, one associated with each of (31) and (32). In other words, the number of state variables describing the dynamic behaviors of synapses is reduced from two per synapse to two per postsynaptic neuron. For LSMs that are comprised of hundreds of neurons and at least thousands of synapses, this merging techniques reduces the effective number of synaptic state variables by at least one order of magnitude.

C. Precision of Synaptic Weights

Due to the large number of synapses, the number of bits used to represent each synaptic weight also has a considerable influence on the hardware overhead. Using a less number of bits reduces the hardware cost but limits the precision of synaptic weights. For example, reducing the precision of synaptic weights may immediately impact the precision in adjusting the location and orientation of the hyperplane of each linear classifier, as shown in Fig. 4, thereby potentially degrading the LSM performance. The detailed tradeoffs between the number of synaptic bits and network performance are studied in the next section.

VII. Experimental Results

In this section, we first introduce the settings of our experiments. To reduce hardware resources of potential implementations, we study the impacts of the bitwidth of synaptic weights and reservoir size on network performance. Finally, our LSM are compared against the existing work on speech recognition. The presented simulation results are based on a C++ simulation program running on a single-core 2.2-GHz AMD Opteron 6174 processor.

A. Experimental Settings

The proposed LSM is set up using parameters summarized in Tables I–III for parameter values used in the LIF neuron model, synaptic connectivity, and learning rule, respectively.

TABLE I
PARAMETER VALUES IN LIF NEURON MODEL

Parameters in LIF model	Value
Threshold voltage V_{th}	$20\ mv$
Resting potential V_{th}	$0\ mv$
Time constant τ_m	$32\ ms$
Time constant τ_c	$64\ ms$
Refractory period τ_{refrac}	$2\ ms$

TABLE II
PARAMETER VALUES IN SYNAPTIC MODEL

Parameters in synaptic model	Value	
r (in Eqn. (3))	2	
k (in Eqn. (3))	type	value
	$E \to E$	0.45
	$E \to I$	0.3
	$I \to E$	0.6
	$I \to I$	0.15
fixed synaptic weight in the reservoir	type	value
	$E \to E$	3
	$E \to I$	6
	$I \to E$	-2
	$I \to I$	-2
W_{max}	type	value
	$E \to E/I$	$8(1-2^{n_{bit}-1})$
	$I \to E/I$	8
W_{min}	type	value
	$E \to E/I$	-8
	$I \to E/I$	$-8(1-2^{n_{bit}-1})$
ΔW	$0.0002 \times 2^{n_{bit}-4}$	
d_{ij}	$1\ ms$	

TABLE III
PARAMETER VALUES IN THE LEARNING RULE

Parameters in the learning rule	Value
C_θ	5
ΔC	3
δ	1

In Table II, E and I denote excitatory and inhibitory neurons, respectively. $E \to I$ denotes the type of synapses with excitatory presynaptic neurons and inhibitory postsynaptic neurons. Note that the time constants of membrane voltage and calcium variable in Table I are set to powers of two for simplified division, as discussed in Section VI-A. In each neuron, 16-bit binary numbers are used for membrane voltage and calcium concentration. Within the reservoir, 20% neurons are inhibitory and 80% are excitatory. The weights of synapses connecting these neurons are fixed. In the LSM, to feed the input signal to the reservoir, each spike train produced in the preprocessing stage is sent to four randomly chosen reservoir neurons through synapses with fixed values randomly chosen to be 8 or -8. To classify the output signals from the reservoir, reservoir neurons are fully connected to each readout neuron by plastic synapses, which are trained by the proposed learning rule.

To evaluate the performance of the proposed LSM, we use three subsets of TI46 speech corpus as benchmarks. The first is widely used in several related existing works, which contains isolated word utterances of five different speakers. Ten different utterances of each word in the 10-word set of zero, two, ..., and nine have been recorded for each speaker. Thus, the benchmark contains 500 speech samples. This is also the main benchmark used to evaluate the performance of the proposed LSM in this paper except for the results presented in Sections VII-D and VII-E for which the following two subsets are used. The second subset includes all digit utterances in the TI46 speech corpus. This large subset contains 1590 recorded speech samples from 16 speakers, eight males and eight females. For each speaker, there are about 10 recordings of each spoken digit. The third subset contains 10 utterances of each letter from *A* to *Z* from a single speaker with a total of 260 speech samples.

We adopt a five-fold cross validation scheme to test the LSM performance. In this setup, all samples in the benchmark are divided into five groups: G_1, G_2, G_3, G_4, and G_5, based on which five different LSMs are trained and tested. For the kth ($k = 1, 2, 3, 4, 5$) LSM, the testing dataset is group G_k and training dataset is the union of all other groups. During the training of an LSM, the temporal signal (speech) is applied to the system shown in Fig. 3. In the meantime, a teacher signal is applied to each readout neuron. After the preprocessing, the input signal is transformed into 77 spike trains that are fed into the reservoir. After applying each training sample to the LSM once, the LSM is trained for one *epoch*. To optimize the LSM performance, the LSM is trained for multiple epochs. During the testing phase, at a time, a testing speech sample is applied to the network while no teacher signal is applied to any readout neuron. The recognition decision is based on the activities of readout neurons and made at the end of each testing speech sample. The neuron that has fired the greatest number of spikes is the winner, whose associated class label is deemed to be the classification decision of the LSM.

B. Precision of Synaptic Weights

As discussed in Section VI-B, the number of synapses in the LSM is much larger than that of neurons. Efficient realization of synaptic models is important for minimizing the overall hardware cost of the system. We study the influence of the bitwidth (precision) of synaptic weights on the LSM performance. The reservoir size used is $3 \times 3 \times 15$ [8].

Fig. 11 shows the performances of the LSM with different precisions of synaptic weights. The influence of the bitwidth of synaptic weights is summarized in Table IV, where the initial results and best results are shown for synapses using different bitwidths. The initial results are obtained by training the LSM for only one epoch. The best results represent the highest performance levels achieved after the LSM is trained for 500 epochs. To reduce the performance fluctuations over different epochs, as observed in Fig. 11, the best performance is the averaged recognition rate over a 20-epoch interval. Note that in Table IV, the sum of each recognition rate and the corresponding error rate may be less than 100%. This is because the network may not be able to recognize a small number of speech samples when more than one readout neuron have fired the largest number of spikes. Since in this case these input samples may be further recognized by an additional classifier, the situation is better than misclassification.

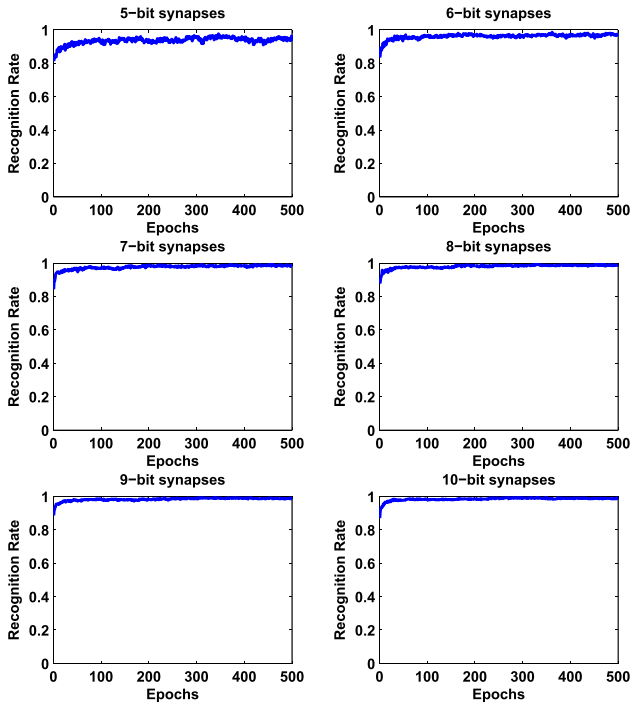


Fig. 11. Influence of the precision of synaptic weights on the LSM performance as synaptic weight resolution varies between 5 and 10 bits. The LSM performances for the first 500 epochs of training are evaluated using the five-fold cross validation with 500 speech samples. The network performance generally improves as the number of bits increases and is saturated at a resolution of 8 bits.

TABLE IV
PERFORMANCE OF LSM VERSUS BITWIDTH OF SYNAPTIC WEIGHTS

Bitwidth of synaptic weights	No. of correctly recognized inputs		Recognition rate		Error rate	
	Initial	Best	Initial	Best	Initial	Best
5	407	480.5	81.4%	96.09%	17.0%	3.24%
6	417	487.5	83.4%	97.49%	12.4%	2.00%
7	422	493.9	84.4%	98.77%	13.0%	0.96%
8	439	494.8	87.8%	98.96%	7.8%	0.75%
9	443	495.2	88.6%	99.03%	5.4%	0.86%
10	435	495.5	87.0%	99.09%	5.4%	0.80%

It is clear that the performance of the LSM generally increases with the bitwidth of synaptic weights. The LSM performances improve rapidly at the beginning of the training process. However, the performance almost saturates when the bitwidth of synaptic weights exceeds eight. This implies that with a resolution of 8-bits or more, the precision of synaptic weights is no longer a performance-limiting factor. This observation provides practical design guidance for maintaining a good performance level without overdesign.

Training and testing of the five recognizers in the five-fold cross validation based upon 500 epochs takes about 25.8-h wall-clock time, or 5.16 h for each recognizer. On average, the training and testing of each recognizer for one epoch takes only about 37 s in total.

C. Size of the Reservoir

We examine the recognition rate and error rate as functions of the size of the reservoir. We use the same settings and parameter values introduced at the beginning of this section and 10-bit binary numbers for synaptic weights. In Table V, the first and second columns show the shape and the size of the reservoir. The third and fourth columns show the recognition rates and error rates, respectively. Among these tested LSMs, the best achieved performance has a recognition rate of 99.79% and an error rate of 0.08%.

TABLE V
PERFORMANCE OF LSM WITH VARIOUS RESERVOIR SIZES

Reservoir shape	Reservoir size	Recognition rate	Error rate
$2 \times 2 \times 20$	80	96.25%	3.24%
$2 \times 2 \times 30$	120	99.31%	0.41%
$2 \times 2 \times 40$	160	98.74%	1.16%
$2 \times 2 \times 50$	200	99.21%	0.58%
$3 \times 3 \times 10$	90	97.16%	2.18%
$3 \times 3 \times 15$	135	99.09%	0.80%
$3 \times 3 \times 20$	180	98.78%	1.01%
$3 \times 3 \times 25$	225	98.87%	1.01%
$3 \times 3 \times 30$	270	98.55%	1.37%
$3 \times 3 \times 40$	360	98.90%	0.88%
$3 \times 3 \times 50$	450	98.91%	0.92%
$4 \times 4 \times 5$	80	96.90%	2.97%
$4 \times 4 \times 10$	160	98.43%	1.44%
$4 \times 4 \times 15$	240	98.69%	1.29%
$4 \times 4 \times 20$	320	99.08%	0.74%
$4 \times 4 \times 25$	400	98.15%	1.78%
$5 \times 5 \times 5$	125	99.10%	0.81%
$5 \times 5 \times 10$	250	99.00%	1.00%
$5 \times 5 \times 15$	375	98.54%	1.27%
$6 \times 6 \times 6$	216	99.24%	0.61%
$6 \times 6 \times 8$	288	98.79%	1.11%
$6 \times 6 \times 10$	360	99.64%	0.28%
$6 \times 6 \times 12$	432	98.86%	0.92%
$7 \times 7 \times 7$	343	99.79%	0.08%

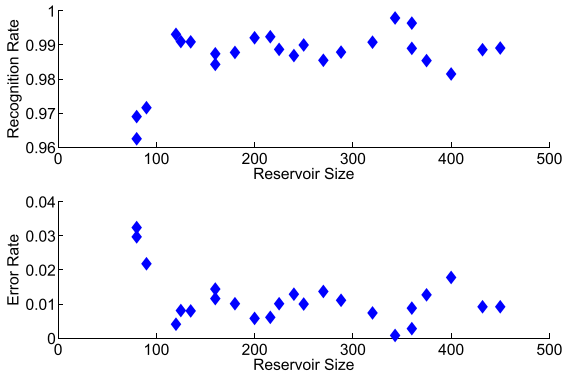


Fig. 12. Recognition and error rates of the LSM as functions of the size (number of neurons) of the reservoir.

Clearly, the LSM performance varies with reservoir size. The relation between them is visualized in Fig. 12. From the figure, it can be observed that when reservoir size is less than 100 neurons, the LSM performance is significantly lower than those for the cases with a larger reservoir. When the reservoir size is 100 neurons or larger, the recognition and error rates are around 99% and 1%, respectively. Further increasing the reservoir size beyond 100 neurons does not significantly boost the LSM performance. Therefore, considering both the performance and the cost effectiveness, a reservoir of size slightly larger than 100 neurons can be a good choice for this application. To further verify this observation, we fix the

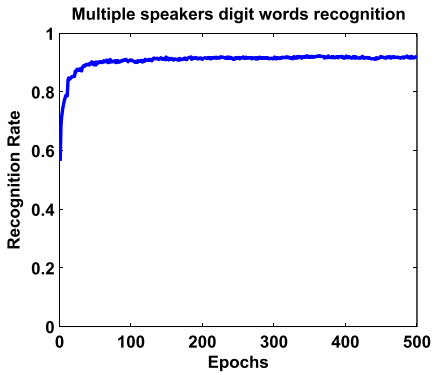


Fig. 13. Classification performance of the proposed LSM on the 1590-sample dataset containing all digit utterances in TI46 speech corpus. The final classification rate reaches 92.3%.

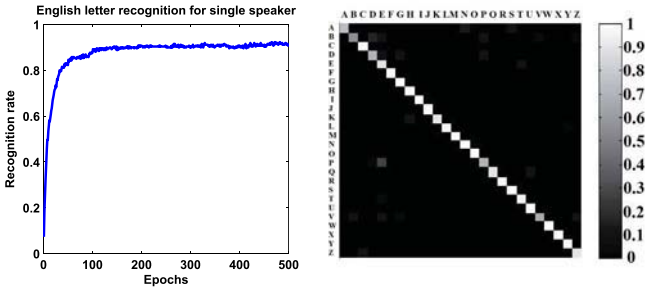


Fig. 14. Left: the English letter classification performance as a function of the number of training epochs. The final LSM classification rate is 92.3%. Right: classification performance on each letter. The row and column indices represent the true class and output of the LSM, respectively. For example, the grayscale square with row index E and column index F shows the probability of letter E to be misclassified as F. The proposed LSM classifies most letters perfectly.

reservoir size to $3 \times 3 \times 15$ with 135 neurons, but generate four LSMs with different randomly generated synapses within the reservoir and between the input neurons and the reservoir neurons. The classification rates are 97.97%, 97.93%, 98.04%, and 97.88%, respectively, for these four cases.

D. Classification of Multispeaker Spoken Digits

To test our approach on larger test cases, we adopt an additional subset of the TI46 speech corpus, which is the second benchmark described in Section VII-A. This is a larger multispeaker isolated digit subset with 1590 speech samples. The LSM used has 83 input neurons, 135 reservoir neurons, 10 readout neurons, a synaptic weight resolution of 10 bits, and a membrane voltage resolution of 16 bits. For this 1590-sample benchmark, the final classification rate of the proposed LSM is 92.3% as shown in Fig. 13.

E. Classification of English Letters

To test the LSM performance with increased numbers of classes, we apply the proposed LSM to the third benchmark described in Section VII-A, which has 260 samples of utterances of English letters A to Z. The LSM used here has 83 input neurons, 135 reservoir neurons, 26 readout neurons, a synaptic weight resolution of 10 bits, and a membrane voltage resolution of 16 bits. As shown in Fig. 14, the final performance is 92.3% with most letters classified perfectly. A significantly worse performance takes place only on a few letters with similar pronunciations such as letters B and D, which may be difficult to be distinguished by the humans without full attention. Compared with the 10-digit utterance dataset, these results imply that the bottleneck to further LSM performance improvement on the 26-letter utterance dataset is the difficulty in distinguishing a few highly similar input classes rather than the increased number of classes.

F. Comparison With Other Methods

Since the 10-class digit subset of TI46 is widely applied to test the LSM performance, while the 26-class English letter subset is rarely used in the literature, we only compare our approach with other related methods based on the former. Systematic comparison between different methods is, in general, a difficult task since the achieved performance may depend on both the learning algorithm adopted and specific experimental setups used, the latter of which may vary across reported works and are often not fully reported. Nevertheless, we use the final classification rate as the only performance measure in the comparison.

Verstraeten *et al.* [19] used the same TI46 subset and also experimental setups (such as the preprocessing module introduced in Section II-B) similar to this paper. The main difference between the two works lies in the training method used. Verstraeten *et al.* [19] adopted ridge regression for training the plastic synapse of readout neurons. With the reservoir size ranging from 300 to 2000 neurons, the word error rate (WER) on the testing dataset reported in [19] is between 10% and 3%, which is significantly higher than that of the proposed LSM.

Ghani *et al.* [20] used different LSM parameter settings with a much smaller reservoir of 8 to 27 neurons. The speech samples from the TI46 corpus were preprocessed by temporal-based linear predictive coding. A complicated multilayer feedforward network consisting of 62 to 168 neurons trained using standard backpropagation algorithms was used for backend processing. This work reported recognition rates between 80% and 100%. Ghani *et al.* [20] used only a total of 200 speech samples, which were divided into two datasets for training and testing, respectively. The use of small datasets in this case may lead to greater performance variations. In addition, [20] only reported the best result from different trials for each network configuration.

Graves *et al.* [24] proposed long short-term memory recurrent neural networks for speech recognition. Their network was constructed by connecting 121 units of different types with the MFCC method used for signal preprocessing. Their reported WER on the TI46 benchmark was 2%.

The state-of-the-art HMM based recognizer Sphinx-4 was introduced in [21] (a previous version Sphinx-2 has been adopted in commercial products). Tested on the TI46 data set, a WER of 0.168% was reported. Comparison between this WER and our best error rate of 0.08% error rate and recognition rate of 99.79% shows the top-notch performance of the proposed LSM. Note that Sphinx-4 used dataset-specific

language and acoustic models and was heavily tuned for the specific dataset.

VIII. Conclusion

We present a bioinspired digital LSM for low-power VLSI-based machine learning applications, such as speech recognition. The proposed spike-based online learning rule is local, has low algorithmic complexity, and facilitates VLSI implementation by avoiding communications between nonneighboring elements of the neural network. Using speech samples from the TI46 speech corpus as the benchmark, we study the influence of synaptic dynamics on the LSM performance and present techniques to greatly reduce the hardware implementation cost of the LSM. To improve the cost effectiveness of our proposed technique, we also study the tradeoffs between the number of bits used for synaptic weights, the size of the reservoir, and the resulting LSM performance. Tested on a subset of TI46 speech corpus, our proposed LSM is demonstrated to have a top-notch performance among several related speech recognizers including the state-of-the-art HMM-based Sphinx-4 recognizer.

References

[1] P. Dayan and L. F. Abbott, *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. Cambridge, MA, USA: MIT Press, 2001.

[2] D. Bush and Y. Jin, "Calcium control of triphasic hippocampal STDP," *J. Comput. Neurosci.*, vol. 33, no. 3, pp. 495–514, Dec. 2012.

[3] S. Honnuraiah and R. Narayanan, "A calcium-dependent plasticity rule for HCN channels maintains activity homeostasis and stable synaptic learning," *PLoS ONE*, vol. 8, no. 2, p. e55590, 2013.

[4] W. Mass, "Networks of spiking neurons: The third generation of neural network models," *Neural Netw.*, vol. 10, no. 9, pp. 1659–1671, 1997.

[5] S. M. Bohte, J. N. Kok, and H. L. Poutre, "SpikeProp: Backpropagation for networks of spiking neurons," in *Proc. 8th Eur. Symp. Artif. Neural Netw.*, Bruges, Belgium, Apr. 2000, pp. 419–424.

[6] T. Natschlager and B. Ruf, "Spatial and temporal pattern analysis via spiking neurons," *Netw., Comput. Neural Syst.*, vol. 9, no. 3, pp. 319–332, 1998.

[7] J. J. Wade, L. J. McDaid, J. A. Santos, and H. M. Sayers, "SWAT: A spiking neural network training algorithm for classification problems," *IEEE Trans. Neural Netw.*, vol. 21, no. 11, pp. 1817–1830, Nov. 2010.

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

[9] J. M. Brader, W. Senn, and S. Fusi, "Learning real-world stimuli in a neural network with spike-driven synaptic dynamics," *Neural Comput.*, vol. 19, no. 11, pp. 2881–2912, 2007.

[10] J. L. McKinstry and G. M. Edelman, "Temporal sequence learning in winner-take-all networks of spiking neurons demonstrated in a brain-based device," *Frontiers Neurorobot.*, vol. 7, no. 10, pp. 1–10, 2013.

[11] Y. Meng, Y. Jin, and J. Yin, "Modeling activity-dependent plasticity in BCM spiking neural networks with application to human behavior recognition," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 1952–1966, Dec. 2011.

[12] J. Misra and I. Saha, "Artificial neural networks in hardware: A survey of two decades of progress," *Neurocomputing*, vol. 74, nos. 1–3, pp. 239–255, 2010.

[13] B. Brezzo *et al.*, "A 45 nm CMOS neuromorphic chip with a scalable architecture for learning in networks of spiking neurons," in *Proc. IEEE Custom Integr. Circuits Conf.*, San Jose, CA, USA, Sep. 2011, pp. 1–4.

[14] P. Merolla, J. Arthur, F. Akopyan, N. Imam, R. Manohar, and D. S. Modha, "A digital neurosynaptic core using embedded crossbar memory with 45 pJ per spike in 45 nm," in *Proc. IEEE Custom Integr. Circuits Conf.*, San Jose, CA, USA, Sep. 2011, pp. 1–4.

[15] Y. Kim, Y. Zhang, and P. Li, "A digital neuromorphic VLSI architecture with memristor crossbar synaptic array for machine learning," in *Proc. IEEE Int. SOC Conf.*, Niagara Falls, NY, USA, Sep. 2012, pp. 328–333.

[16] W. Maass, T. Natschlager, and H. Markram, "Computational models for generic cortical microcircuits," in *Computational Neuroscience: A Comprehensive Approach*, J. Feng, Ed. Boca Raton, FL, USA: Chapman & Hall, 2004.

[17] M. J. Embrechts, L. A. Alex, and J. D. Linton, "Reservoir computing for static pattern recognition," in *Proc. 17th Eur. Symp. Artif. Neural Netw.*, Bruges, Belgium, Apr. 2009, pp. 245–250.

[18] Y. Zhang and K. Wang, "The application of liquid state machines in robot path planning," *J. Comput.*, vol. 4, no. 11, pp. 1182–1186, 2009.

[19] D. Verstraeten, B. Schrauwen, D. Stroobandt, and J. Van Campenhout, "Isolated word recognition with the *Liquid State Machine*: A case study," *Inf. Process. Lett.*, vol. 95, no. 6, pp. 521–528, 2005.

[20] A. Ghani, T. M. McGinnity, L. Maguire, L. McDaid, and A. Belatreche, "Neuro-inspired speech recognition based on reservoir computing," in *Advances in Speech Recognition*, N. Shabtai, Ed. Rijeka, Croatia: InTech, 2010.

[21] W. Walker *et al.*, "Sphinx-4: A flexible open source framework for speech recognition," Sun Microsystems Inc., Mountain View, CA, USA, Tech. Rep. SMLI TR-2004-139, Nov. 2004.

[22] R. K. Moore, "Twenty things we still don't know about speech," in *Proc. CRIM/FORWISS Workshop Progr. Prospects Speech Res. Technol.*, Munich, Germany, Sep. 1994, pp. 9–17.

[23] M. A. Anusuya and S. K. Katti, "Speech recognition by machine: A review," *Int. J. Comput. Sci. Inf. Security*, vol. 6, no. 3, pp. 181–205, 2009.

[24] A. Graves, D. Eck, N. Beringer, and J. Schmidhuber, "Biologically plausible speech recognition with LSTM neural nets," in *Biologically Inspired Approaches to Advanced Information Technology*, A. Ijspeert, M. Murata, and N. Wakamiya, Eds. Berlin, Germany: Springer-Verlag, 2004.

[25] B. Schrauwen, M. D'Haene, D. Verstraeten, and J. Van Campenhout, "Compact hardware liquid state machines on FPGA for real-time speech recognition," *Neural Netw.*, vol. 21, no. 2, pp. 511–523, 2008.

[26] T. Netschlager, W. Maass, and H. Markram, "The 'liquid computer': A novel strategy for real-time computing on time series," *Found. Inf. Process. TELEMATIK*, vol. 8, no. 1, pp. 39–43, 2002.

[27] T. M. Cover, "Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition," *IEEE Trans. Electron. Comput.*, vol. EC-14, no. 3, pp. 326–334, Jun. 1965.

[28] J. J. Hopfield and C. D. Brody, "What is a moment? 'Cortica' sensory integration over a brief interval," *Proc. Nat. Acad. Sci. United States Amer.*, vol. 97, no. 25, pp. 13919–13924, 2000.

[29] M. Xu, L.-Y. Duan, J. Cai, L.-T. Chia, C. Xu, and Q. Tian, "HMM-based audio keyword generation," in *Advances in Multimedia Information Processing*, K. Aizawa, Y. Nakamura, and S. Satoh, Eds. Berlin, Germany: Springer-Verlag, 2005.

[30] M. Sahidullah and G. Soha, "Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition," *Speech Commun.*, vol. 54, no. 4, pp. 543–565, 2012.

[31] R. F. Lyon, "A computational model of filtering, detection, and compression in the cochlea," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, Palo Alto, CA, USA, May 1982, pp. 1282–1285.

[32] L. M. Van Immerseel and J. P. Martens, "Pitch and voiced/unvoiced determination with an auditory model," *J. Acoust. Soc. Amer.*, vol. 91, no. 6, pp. 3511–3526, 1993.

[33] B. Schrauwen and J. Van Campenhout, "BSA, a fast and accurate spike train encoding scheme," in *Proc. IEEE Int. Joint Conf. Neural Netw.*, Portland, OR, USA, Jul. 2003, pp. 2825–2830.

[34] M. Slaney, "Lyon's cochlea model," Apple Computer, Inc., Cupertino, CA, USA, Tech. Rep. 13, 1988.

[35] D. O. Hebb, *The Organization of Behavior*. New York, NY, USA: Wiley, 1949.

[36] E. L. Bienenstock, L. N. Cooper, and P. W. Munro, "Theory for the development of neuron selectivity: Orientation specificity and binocular interaction in visual cortex," *J. Neurosci.*, vol. 2, no. 1, pp. 32–48, 1982.

[37] P. I. Good and J. W. Hardin, *Common Errors in Statistics (and How to Avoid Them)*, 4th ed. Hoboken, NJ, USA: Wiley, 2012.

[38] N. Brunel, F. Carusi, and S. Fusi, "Slow stochastic Hebbian learning of classes of stimuli in a recurrent neural network," *Comput. Neural Syst.*, vol. 9, no. 1, pp. 123–152, 1998.

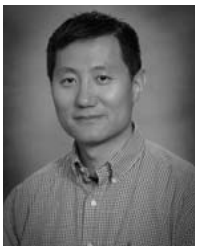
[39] W. Senn and S. Fusi, "Learning only when necessary: Better memories of correlated patterns in networks with bounded synapses," *Neural Comput.*, vol. 17, no. 10, pp. 2106–2138, 2005.

[40] W. Senn and S. Fusi, "Convergence of stochastic learning in perceptrons with binary synapses," *Phys. Rev. E*, vol. 71, no. 6, pp. 061907-1–061907-12, 2005.

[41] R. Gütig and H. Sompolinsky, "The tempotron: A neuron that learns spike timing-based decisions," *Nature Neurosci.*, vol. 9, no. 3, pp. 420–428, 2006.
[42] F. Ponulak and A. Kasinski, "Supervised learning in spiking neural networks with ReSuMe: Sequence learning, classification, and spike shifting," *Neural Comput.*, vol. 22, no. 2, pp. 467–510, 2010.
[43] R. V. Florian, "The chronotron: A neuron that learns to fire temporally precise spike patterns," *PLoS ONE*, vol. 7, no. 8, p. e40233, 2012.
[44] A. Mohemmed, S. Schliebs, S. Matsuda, and N. Kasabov, "Span: Spike pattern association neuron for learning spatio-temporal spike patterns," *Int. J. Neural Syst.*, vol. 22, no. 4, p. 1250012, 2012.
[45] B. Widrow and M. E. Hoff, "Adaptive switching circuits," in *Proc. IRE WESCON Conv. Rec.*, Aug. 1960, pp. 96–104.
[46] N. Bertschinger and T. Natschlager, "Real-time computation at the edge of chaos in recurrent neural networks," *Neural Comput.*, vol. 16, no. 7, pp. 1413–1436, 2004.
[47] R. Legenstein and W. Maass, "What makes a dynamical system computationally powerful?" in *New Directions in Statistical Signal Processing: From Systems to Brains*, S. Haykin, J. C. Principe, T. J. Sejnowski, and J. G. McWhirter, Eds. Cambridge, MA, USA: MIT Press, 2007.
[48] R. Legenstein and W. Maass, "Edge of chaos and prediction of computational performance for neural circuit models," *Neural Netw.*, vol. 20, no. 3, pp. 323–334, 2007.
[49] D. V. Buonomano and W. Maass, "State-dependent computations: Spatiotemporal processing in cortical networks," *Nature Rev. Neurosci.*, vol. 10, pp. 113–125, Feb. 2009.
[50] M. Tsodyks and S. Wu, "Short-term synaptic plasticity," *Scholarpedia*, vol. 8, no. 10, p. 3153, 2013.
[51] N. H. E. Weste and D. M. Harris, *CMOS VLSI Design: A Circuits and Systems Perspective*, 3rd ed. Boston, MA, USA: Addison-Wesley, 2004.
[52] W. Cui, H. Chen, and Y. Han, "VLSI implementation of universal random number generator," in *Proc. IEEE Asia-Pacific Conf. Circuits Syst.*, Denpasar, Indonesia, Oct. 2002, pp. 456–470.
[53] C. Bernard, Y. C. Ge, E. Stockley, J. B. Willis, and H. V. Wheal, "Synaptic integration of NMDA and non-NMDA receptors in large neuronal network models solved by means of differential equations," *Biol. Cybern.*, vol. 70, no. 3, pp. 267–273, 1994.
[54] Y. Zhang, B. Yan, M. Wang, J. Hu, and P. Li, "Linking brain behavior to underlying cellular mechanisms via large-scale brain modeling and simulation," *Neurocomputing*, vol. 97, pp. 317–331, Nov. 2012.

Yong Zhang received the B.S. degree in engineering from the Department of Electronic Engineering, Shanghai Jiao Tong University, Shanghai, China, in 2008, and the Ph.D. degree in computer engineering from Texas A&M University, College Station, TX, USA, in 2014. He is currently an R&D engineer with Cadence Design Systems, San Jose, CA, USA.

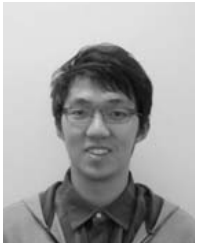
His current research interests include simulation and analysis of very large scale integrated, biologically-motivated computing systems, and biological systems.



Peng Li (S'02–M'04–SM'09) received the Ph.D. degree in electrical and computer engineering from Carnegie Mellon University, Pittsburgh, PA, USA, in 2003.

He is currently an Associate Professor with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA. His current research interests include the general areas of circuits and systems, electronic design automation, and computational neuroscience.

Prof. Li has been recognized by various awards, including the IEEE/ACM William J. McCalla ICCAD Best Paper Award, three IEEE/ACM Design Automation Conference Best Paper Awards, two SRC Inventor Recognition Awards, two MARCO Inventor Recognition Awards, and the National Science Foundation CAREER Award. He was a recipient of the ECE Outstanding Professor Award from Texas A&M University and was named as a TEES Fellow and William O. and Montine P. Head Faculty Fellow. He was an Associate Editor of the IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS from 2008 to 2013. He is also an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-II: EXPRESS BRIEFS. He has served on the committees of many international conferences and workshops.



Yingyezhe Jin received the B.S. degree in electronic and information engineering from Zhejiang University, Hangzhou, China, in 2013. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, Texas A&M University, College Station, TX, USA.

His current research interests include simulation and analysis of bio-inspired neural networks and hardware implementation of neural networks.

Yoonsuck Choe (M'06–SM'14) is currently a Professor and the Director of the Brain Networks Laboratory with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX, USA. His current research interests include computational neuroscience, computational neuroanatomy, neuroinformatics, neural networks, and neuroevolution. His work ranges from visual cortical modeling, sensorimotor learning, temporal aspects of brain function (delay, memory, and prediction), whole brain physical sectioning imaging (knife-edge scanning microscopy), and web-based brain atlas frameworks.