An Online Unsupervised Structural Plasticity Algorithm for Spiking Neural Networks

Subhrajit Roy, *Student Member, IEEE*, and Arindam Basu, *Member, IEEE*

***Abstract*— In this paper, we propose a novel winner-take-all (WTA) architecture employing neurons with nonlinear dendrites and an online unsupervised structural plasticity rule for training it. Furthermore, to aid hardware implementations, our network employs only binary synapses. The proposed learning rule is inspired by spike-timing-dependent plasticity but differs for each dendrite based on its activation level. It trains the WTA network through formation and elimination of connections between inputs and synapses. To demonstrate the performance of the proposed network and learning rule, we employ it to solve two-class, four-class, and six-class classification of random Poisson spike time inputs. The results indicate that by proper tuning of the inhibitory time constant of the WTA, a tradeoff between specificity and sensitivity of the network can be achieved. We use the inhibitory time constant to set the number of subpatterns per pattern we want to detect. We show that while the percentages of successful trials are 92%, 88%, and 82% for two-class, four-class, and six-class classification when no pattern subdivisions are made, it increases to 100% when each pattern is subdivided into 5 or 10 subpatterns. However, the former scenario of no pattern subdivision is more jitter resilient than the later ones.**

***Index Terms*— Spike-timing-dependent plasticity, spiking neural networks, structural plasticity, unsupervised learning, winner-take-all.**

I. Introduction and Motivation

THE winner-take-all (WTA) is a computational framework in which a group of recurrent neurons cooperate and compete with each other for activation. The computational power of WTA [1]–[3] and its function in cortical processing [1], [4] have been studied in detail. Various models and hardware implementations of WTA have been proposed for both rate [5]–[12] and spike-based [13]–[15] neural networks. In recent past, researchers have looked into the application of spike-timing-dependent plasticity (STDP) learning rule on WTA circuits. The performance of competitive spiking neurons trained with STDP has been studied for different types of input, such as discrete spike volleys [16]–[18], periodic inputs [19], [20], and inputs with random intervals [15], [21], [22].

Manuscript received June 12, 2015; revised December 2, 2015 and February 10, 2016; accepted June 8, 2016. Date of publication July 9, 2016; date of current version March 15, 2017. This work was supported by the Ministry of Education, Singapore, under Grant MOE ARC 8/13 and Grant RG 21/10.

The authors are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: subhrajit.roy@ntu.edu.sg; arindam.basu@ntu.edu.sg).

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.2016.2582517

In this paper, for the first time, we are proposing a WTA network, which uses neurons with nonlinear dendrites (NNLD) and binary synapses as the basic computational units. This architecture, which we refer to as WTA employing NNLS (WTA-NNLD), uses a novel branch-specific STDP-based network rewiring (STDP-NRW) learning rule for its training. We have earlier presented [23] a branch-specific STDP rule for batch learning of a supervised classifier constructed of NNLDs. The primary differences of our current approach with [23] are as follows.

1) We present an unsupervised learning rule for training a WTA network.
2) We propose an online learning scheme where connection modifications occur after the presentation of each pattern.

In this paper, we consider spike train inputs with patterns occurring at a random order, which is the same type presented in [15]. The primary differences between this paper and the one proposed in [15] are as follows.

1) Our WTA network is composed of NNLD instead of traditional neurons with no dendrites.
2) Unlike the network proposed in [15] that requires high-resolution weights, the proposed network uses low-resolution nonnegative integer weights and trains itself through modifying connections of inputs to dendrites. Hence, the change of the morphology or structure of the neurons (in terms of connectivity pattern) reflects the learning. This results in easier hardware implementation, since a low-resolution nonnegative integer weight of W can be implemented by activating a shared binary synapse W times through time multiplexing schemes, such as address event representation [24], [25].
3) In [15], though the neurons were allowed to learn and respond to subpatterns, there was no actual guideline or control parameter to set the number of subpatterns to be learned. Here, we utilize the slow time constant of the inhibitory signal to select the number of subpatterns we want to divide a pattern into.

In Section II, we will present an overview of NNLD, propose the WTA-NNLD architecture and STDP-NRW learning rule, and show how the inhibitory slow time constant can be used to select subpatterns within a pattern. Then, we shall provide guidelines on selecting the parameters associated with WTA-NNLD and STDP-NRW. In Section IV, we will describe the classification task considered in this paper, which will be followed by the results. We will also present the robustness of the proposed method to the variations of parameters in

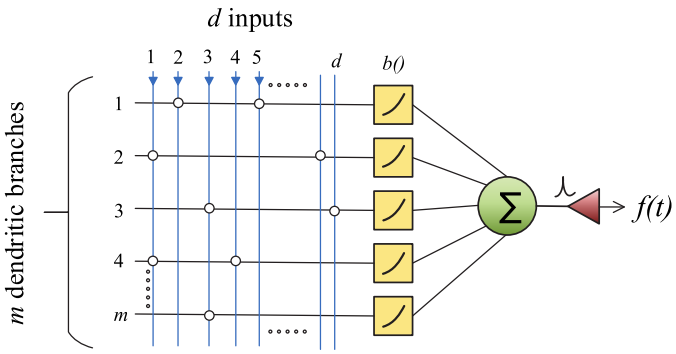

Fig. 1. Neuronal cell with nonlinear dendrites.

Section V, a quality that is essential for its implementation in low-power, subthreshold neuromorphic designs that are plagued with mismatch. We will conclude this paper by discussing the implications of our work and future directions in Section VI.

II. Background and Theory

In this section, we shall first present the working principle of an NNLD. This will be followed by a description of the WTA-NNLD architecture and STDP-NRW learning rule. Finally, we will describe the role of the inhibitory time constant in balancing the specificity and sensitivity of the network.

A. Neuron With Nonlinear Dendrites

The computational model of NNLD was first proposed by Poirazi and Mel [26] where they showed that such neurons have higher storage capacity than their nondendritic counterparts. They used two such NNLDs to construct a supervised classifier and demonstrated its performance in pattern memorization. Recently, NNLD has also been employed to develop computationally powerful rate [27] and spike-based [28], [29] supervised classifiers. The structure of NNLD is isomorphic to a feedforward spiking neural network with a single layer of hidden neurons and one output neuron [30]. The lumped dendritic nonlinearities $b()$ are equivalent to the hidden neurons interposed between the input and output layers. However, spiking neurons implement nonlinear thresholding, integration, refractory period, and so on. Hence, it is typically a much larger circuit compared with the square-law nonlinearity of a dendrite, which makes the NNLD an area efficient architecture.

As shown in Fig. 1, an NNLD consists of m dendritic branches having lumped nonlinearities, with each branch containing k excitatory synaptic contact points of weight 1. If we consider a d dimensional input pattern, then each synapse is driven by any one of these input dimensions where $d \gg k$. We use the leaky integrate-and-fire (LIF) model to generate output spikes. Thus, the neuronal membrane voltage is guided by the following differential equation:

$$C\frac{dV(t)}{dt} + \frac{V(t)}{R} = I_{\text{in}}(t)$$
$$\text{If } V(t) \geq V_{\text{thr}}, \quad V(t) \to 0; \ \& f(t) \to 1$$
$$\text{else } f(t) = 0 \tag{1}$$

where $V(t)$, V_{thr}, $I_{\text{in}}(t)$, and $f(t)$ are the membrane voltage, threshold voltage, input current and output spikes of the NNLD, respectively. Let us denote the input spike train arriving at the ith input line as $e^i(t)$, which is given by

$$e^i(t) = \sum_g \delta\left(t - t_g^i\right) \tag{2}$$

where $g = 1, 2, \ldots$ is the label of the spike. Then, the input current $I_{\text{in}}(t)$ to the neuron can be calculated as

$$I_{\text{in}}(t) = \sum_{j=1}^{m} I_{b,\text{out}}^j(t) \tag{3}$$

$$I_{b,\text{out}}^j(t) = b\left(I_{b,\text{in}}^j(t)\right) \tag{4}$$

$$I_{b,\text{in}}^j(t) = \sum_{i=1}^{d} w_{ij}\left(\sum_{t_g^i < t} K\left(t - t_g^i\right)\right) \tag{5}$$

where $b(.)$ is the nonlinear activation function of the dendritic branch characterized by $b(z) = z^2/x_{\text{thr}}$, $I_{b,\text{out}}^j(t)$ is the output current of the jth dendrite, and $I_{b,\text{in}}^j(t)$ is the input current to the dendritic nonlinearity. Here, x_{thr} describes the behavior of the dendritic nonlinear function by setting a limit on the minimum number of coactive synapses required to produce a supralinear response. K denotes the postsynaptic current kernel given by

$$K(t) = I_0\left(e^{-\frac{t}{\tau_s}} - e^{-\frac{t}{\tau_f}}\right) \tag{6}$$

where τ_f and τ_s are the fast and slow time constants governing the rise and fall times, respectively, and I_0 is a normalizing constant. In this paper, we consider low-resolution nonnegative integer weights associated with the input lines. So $w_{ij} \epsilon \{0, 1, \ldots, k\}$ means the number of times the ith input line is connected to the jth dendritic branch. Like our earlier work [29], we allow multiple connections of one input dimension to a single dendrite but restrict the number of connections per dendrite to k by enforcing $\sum_{i=1}^{d} w_{ij} = k$ for each j. The output of the NNLD is a spike train and can be denoted as

$$f(t) = \sum_a \delta(t - t_a) \tag{7}$$

where $a = 1, 2, \ldots$ is the spike index.

B. Winner-Take-All Employing Neurons With Nonlinear Dendrites

We propose a spike-based WTA network, shown in Fig. 2, which is composed of N such NNLDs. Each NNLD is composed of m dendrites, where each dendrite chooses (repetition allowed) k of the d available input lines and connects to k synapses having weight 1. The membrane voltage, threshold voltage, and input current of the nth NNLD are denoted by $V^n(t)$, V_{thr}, and $I_{\text{in}}^n(t)$, respectively, and the dynamics of the membrane voltage is governed by (1). For the nth NNLD, while the presynaptic spike-train arriving at the ith input line is denoted by $e^i(t)$ as before, the emitted output spike train is given by $f^n(t) = \sum_a \delta(t - t_a^n)$. Note that for any applied input pattern, t_a^n is measured from its beginning.

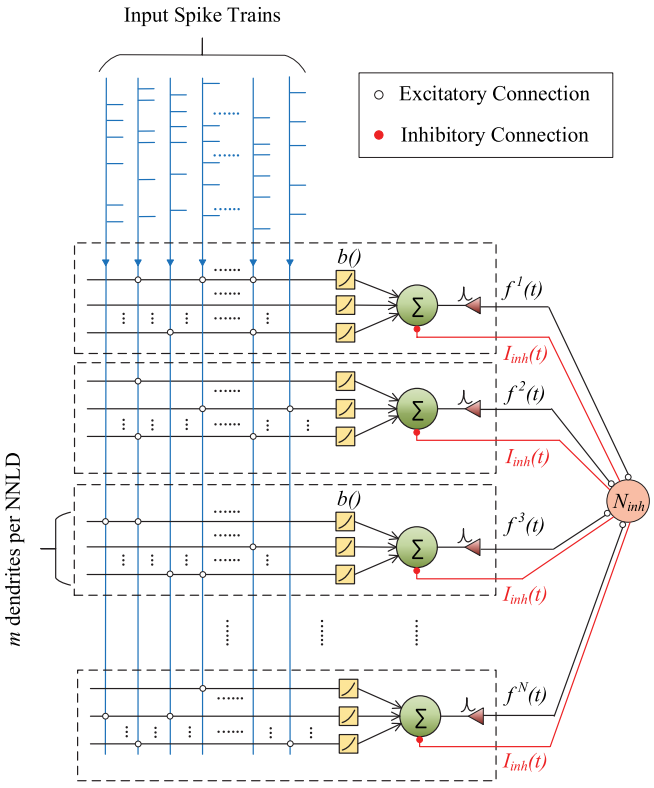

Fig. 2. Spike-based WTA network employing neurons with lumped dendritic nonlinearities as the competing entities. For implementing lateral inhibition, an inhibitory neuron has been included, which, upon activation, provides a global inhibition signal to all the NNLDs.

We have modeled the effect of lateral inhibition by providing each NNLD with a global inhibitory current signal $I_{\text{inh}}(t)$ supplied by a single inhibitory neuron N_{inh} through synapses. The signal $I_{\text{inh}}(t)$ is provided by the inhibitory neuron to all the NNLDs whenever any one of them fires an output spike. $I_{\text{inh}}(t)$ is modeled as $I_{\text{inh}}(t) = K_{\text{inh}}(t - t^n_{\text{last}})$, when the last postsynaptic spike is produced by the nth NNLD at t^n_{last}. The inhibitory postsynaptic kernel, K_{inh}, is given by

$$K_{\text{inh}}(t) = I_{0,\text{inh}}\left(e^{-\frac{t}{\tau_{s,\text{inh}}}} - e^{-\frac{t}{\tau_{f,\text{inh}}}}\right) \tag{8}$$

where $\tau_{f,\text{inh}}$ and $\tau_{s,\text{inh}}$ are the fast and slow time constants dictating the rise and fall times of the inhibitory current, respectively, and $I_{0,\text{inh}}$ sets its amplitude.

C. Spike-Timing-Dependent Plasticity-Based Network Rewiring Learning Rule

Since we consider binary synapses with weight 0 or 1, we do not have the provision to keep real-valued weights associated with them. Hence, to guide the unsupervised learning, we define a correlation coefficient-based fitness value $c^n_{pj}(t)$ for the pth synaptic contact point on the jth dendrite of the nth NNLD of the WTA network, as a substitute for its weight. In the proposed algorithm, structural plasticity or connection modifications happen on longer timescales (at the end of patterns), which is guided by the fitness function $c^n_{pj}(t)$ updated by an STDP inspired rule in shorter timescales

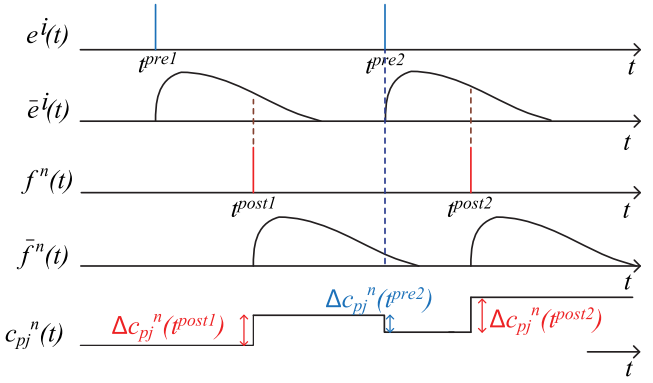

Fig. 3. Example of the update rule of fitness value $[c^n_{pj}(t)]$ is shown. When a postsynaptic spike occurs at t^{post1}, the value of $c^n_{pj}(t)$ increases by $b'_j(t^{\text{post1}})\bar{e}_i(t^{\text{post1}})$. Due to the appearance of a presynaptic spike at t^{pre2}, $c^n_{pj}(t)$ reduces by $b'_j(t^{\text{pre2}})\bar{f}^n(t^{\text{pre2}})$ as shown in the figure.

(at each presynaptic and postsynaptic spike). The operation of the network and learning process comprises the following steps whenever a pattern is presented.

1) $c^n_{pj}(t)$ is initialized as $c^n_{pj}(t = 0) = 0\ \forall p = 1, 2, \ldots k$, $j = 1, 2, \ldots, m$ and $n = 1, 2, \ldots, N$.
2) The value of $c^n_{pj}(t)$ is depressed at presynaptic and potentiated at postsynaptic spikes according to the following rule.
   a) *Depression:* If the presynaptic spike occurs at the pth synapse on the jth dendritic branch of the nth NNLD at time t^{pre}, then the value of $c^n_{pj}(t)$ at $t = t^{\text{pre}}$ is updated by a quantity $\Delta c^n_{pj}(t = t^{\text{pre}})$ given by
   $$\Delta c^n_{pj}(t) = -b'_j(t)\,\bar{f}^n(t)|_{t=t^{\text{pre}}} \tag{9}$$
   where $\bar{f}^n(t) = K(t) * f^n(t)$ is the postsynaptic trace of the nth NNLD and $b'()$ denotes the derivative of the nonlinear function $b()$.
   b) *Potentiation:* If the nth NNLD of the WTA-NNLD network fires a postsynaptic spike at time t^{post}, then $c^n_{pj}(t)$ at $t = t^{\text{post}}\ \forall p = 1, 2, \ldots k$ and $j = 1, 2, \ldots, m$, i.e., for each synapse connected to the nth NNLD is updated by $\Delta c^n_{pj}(t = t^{\text{post}})$, which is given by
   $$\Delta c^n_{pj}(t) = b'_j(t)\bar{e}^i(t)|_{t=t^{\text{post}}} \tag{10}$$
   where $\bar{e}^i(t) = K(t) * e^i(t)$ is the presynaptic trace of the corresponding input line connected to it.

   A pictorial explanation of this update rule of $c^n_{pj}(t)$ is shown in Fig. 3. Note that for a square-law nonlinearity, $b'(z) \propto z$ and hence can be easily computed in hardware without requiring any extra circuitry to calculate the derivative.
3) During the presentation of the pattern whenever a spike is produced by any of the N excitatory NNLDs, the inhibitory neuron N_{inh} sends an inhibitory signal to all the NNLDs of the WTA.

4) After the network has integrated the input spikes over the current pattern duration T_p, the synaptic connections of the NNLDs, which have produced at least one spike are modified.
5) If we consider that Q out of N NNLDs have produced postsynaptic spike/spikes for the current pattern, then the connectivity of the qth NNLD $\forall q = 1, 2 \ldots, Q$ is updated by tagging the synapse ($s^q_{\min}$) having the lowest value of correlation coefficient at $t = T_p$ out of the $m \times k$ synapses connected to it for possible replacement.
6) To aid the unsupervised learning process, randomly chosen sets R^q containing n_R of the d input dimensions are forced to make silent synapses of weight 1 on the dendritic branch of $s^q_{\min}$ $\forall q = 1, 2 \ldots, Q$. We term these synapses as silent, since they do not contribute to the computation of $V^n(t)$—so they do not alter the classification when the same pattern set is reapplied. The value of $c^q_{pj}(t = T_p)$ is calculated for synapses in R^q and the synapse having maximum $c^q_{pj}(t = T_p)$ in R^q denoted by $r^q_{\max}$ $\forall q = 1, 2 \ldots, Q$ is identified. Next, the input line connected to $s^q_{\min}$ is swapped with the input line connected to $r^q_{\max}$. Hence, instead of the traditional method of training by changing of high-resolution synaptic weights, our learning rule modifies the connections between the inputs and dendrites based on the fitness values.
7) All the $c^n_{pj}(t)$ values are reset to zero and the above mentioned steps are repeated whenever a pattern is presented. Here, we define an epoch for C-class classification as a set of patterns consisting of one pattern from each of the C classes in random order. We define another term l_{mean} as the average of the latencies of the postsynaptic spikes in the network over time period of the last epoch, which is given by

$$l_{\text{mean}} = \left\langle \sum_n \sum_a t^n_a \right\rangle_1 \tag{11}$$

where $\langle \cdot \rangle_1$ denotes averaging over one epoch. We note the value of l_{mean} for every epoch and the learning is considered to converge when the value of a convergence measure (CM) based on l_{mean} reaches saturation. We define our CM in Section III.

D. Specificity and Sensitivity: Role of Inhibitory Time Constant

When a pattern is presented to the WTA-NNLD and any one of the N NNLDs produce an output spike, a global inhibition current $I_{\text{inh}}(t)$ is injected into all the N NNLDs. The slow time constant $\tau_{s,\text{inh}}$ of this signal controls the output firing activity of the WTA-NNLD. Typically, a large value of $\tau_{s,\text{inh}}$ (with respect to T_p) is set, and only one NNLD produces an output spike, i.e., patterns of the same class are encoded by a single NNLD. The postsynaptic spike latency for a pattern P is defined as the time difference between the start of the pattern and the first spike produced by any one of the N neurons of WTA-NNLD. During training of WTA-NNLD for this case, different NNLDs get locked onto different classes of pattern and the latency gradually decreases until the end of the training. Thus, after completion of training, the unique NNLDs, which have learned different classes of pattern rely only on the first few spikes (determined by the latency at the end of training) to predict the pattern's class, thereby significantly reducing the prediction time [15]. So, the sensitivity of the network is increased. However, the problems with this approach are as follows.

1) The percentage of successful classifications can be less due to the strict requirement of different neurons firing based only on first few spikes of different patterns (shown in Section IV).
2) Though the prediction time of a pattern's class is significantly reduced, this method neglects most part of the pattern after the first few spikes, which may lead to a lot of false detections.

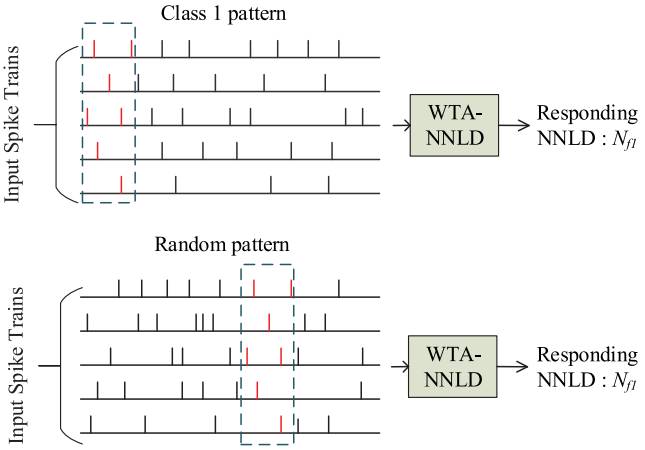

Fig. 4. Specificity is reduced if only one NNLD encodes a pattern based on its first few spikes. As shown, a different pattern with a section resembling the beginning of class 1 pattern may cause neuron N_{f1} to respond.

We demonstrate the limitation mentioned in the above point by a simple example in Fig. 4. Let us consider that we are performing a C-class classification task and assume that after the training phase is complete, NNLD N_{f1} responds to patterns belonging to class 1. NNLD N_{f1} has trained itself to provide an output spike depending on the position of the first few spikes (red spikes in dashed box of Fig. 4) of the pattern. It neglects the rest of the pattern while providing a prediction. However, for longer patterns, there is a chance that this spike set can occur anywhere inside a random pattern (not belonging to any class or to another class). The same NNLD N_{f1} responds to such patterns by producing a postsynaptic spike. Hence, we see that though trained WTA-NNLD is very sensitive in this case, it loses specificity. On the other hand, if we set a moderate value of $\tau_{s,\text{inh}}$, then for a single pattern, multiple NNLDs are capable of producing output spikes. Hence, patterns of the same class are now encoded by a sequence of successive firing of few NNLDs where each NNLD fires for one subpattern. Let n_{sub} be the number of subpatterns that is set by a proper choice of $\tau_{s,\text{inh}}$. Thus, the original case of one NNLD firing for each pattern corresponds to $n_{\text{sub}} = 1$. In this paper, for a C-class classification task, we define a successful trial as one in which 1) during the training phase, WTA-NNLD

learns different unique representations for patterns of different classes and 2) after completion of training and achieving success in 1), the network produces the same representation, when presented with testing patterns corresponding to classes that it had learned during the training phase. When $n_{\text{sub}} = 1$, i.e., no pattern subdivisions are made, this unique representation is a different neuron firing for different classes of patterns. When $n_{\text{sub}} > 1$, the unique representation is a different sequence of successive NNLDs firing for different classes of patterns. When $n_{\text{sub}} > 1$, we allow the NNLDs to detect subpatterns within patterns. Since, in this approach, the WTA-NNLD gives weightage to the entire pattern before predicting its class, the number of false detections can be largely reduced. However, this method has a limitation of being less jitter resilient—one of the many subpatterns can be easily corrupt by noisy jitters in spike (shown in Section IV) and fail to produce a unique identifier during the testing phase. Hence, the choice of n_{sub} and, consequently, the inhibitory time constant depend on the amount of temporal jitter in the application.

III. Choice of Parameters

The following is an exhaustive list of the parameters used by WTA-NNLD and STDP-NRW.

T_p	Duration of a pattern.
d	Dimension of the input.
m	Number of dendrites per NNLD.
k	Number of synapses per dendrite.
n_R	Number of input dimensions in replacement set.
τ_s and τ_f	Slow and fast time constant of excitatory current kernel.
I_0	Normalization constant of excitatory current kernel.
$\tau_{s,\text{inh}}$ and $\tau_{f,\text{inh}}$	Slow and fast time constant of inhibitory current kernel.
$I_{0,\text{inh}}$	Normalization constant of inhibitory current kernel.
x_{thr}	Threshold of dendritic nonlinearity.
V_{thr}	Firing threshold voltage of NNLD.
N	Number of NNLDs in WTA.
C	Number of classes of patterns.

We will now provide some guidelines on choosing the key parameters.

A. Total Number of Synapses per NNLD (s)

The number of synapses allocated to each neuronal cell of WTA-NNLD is kept as equal to the dimension (d) of the input patterns. This is done to ensure NNLD that uses the same amount of synaptic resources as the simplest neuron–a perceptron. Thus, if the proposed network is comprised of N such neuronal cells, then the total number of synaptic resources required is $d \times N$.

B. Number of Dendrites per NNLD (m)

In [26], a measure of the pattern memorization capacity, B_N, of the NNLD (Fig. 1) has been provided by counting all possible functions realizable as

$$B_N = \log_2 \left(\binom{\binom{k+d-1}{k} + m - 1}{m} \right) \text{bits} \tag{12}$$

where m, k, and d are the number of dendrites, number of synapses per dendrites, and dimension of the input, respectively, for this neuronal cell. When a new classification problem is encountered, we first note the value of d, which in turn sets our s, since we have considered $s = d$. Since $s = m \times k$, for a fixed s, all possible values, which m can take are factors of s. We calculate B_N for these values of m by (12). The value of m for which B_N attains its maxima is set as m in our experiment. As an example, we show in Fig. 5 the variation of B_N with m when $d = 100$. It is evident from the curve that the capacity is maximum when $m = 25$ and so in our simulations for classifying 100-D patterns, we employ neuronal cells having 25 dendrites.

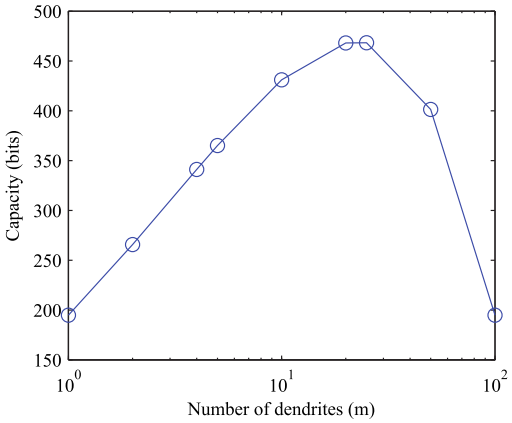

Fig. 5. Pattern memorization capacity of an NNLD (B_N) is plotted as a function of the number of dendrites (m) for a fixed number of input dimensions ($d = 100$) and synapses ($s = 100$).

C. Number of Synapses per Branch (k)

After s and m have been set, the value of k can be computed as $k = (s/m)$.

D. Normalization Constant (I_0), Slow (τ_s), and Fast Time Constant (τ_f) of Excitatory PSC Kernel

The fast time constant (τ_f) and the slow time constant (τ_s) have been defined in Section II-C. In hardware implementation of a synapse [29], τ_f usually takes a small positive value and is typically not tuned. The slow time constant, τ_s, is responsible for integration across temporally correlated spikes and the performance of the network is dependent on its value. If τ_s takes too small a value, then the postsynaptic current due to individual spikes dies down rapidly and thus temporal integration of separated inputs does not take place. On the other hand, large values of τ_s render all spikes effectively simultaneous. So, in both extremes, the extraction of temporal features from the input pattern is hampered. In [29], we have provided a mathematical formula for calculating $\tau_{s,\text{opt}}$, the optimal value of τ_s, with respect to the interspike interval (ISI) of the input pattern for which optimal performance of the

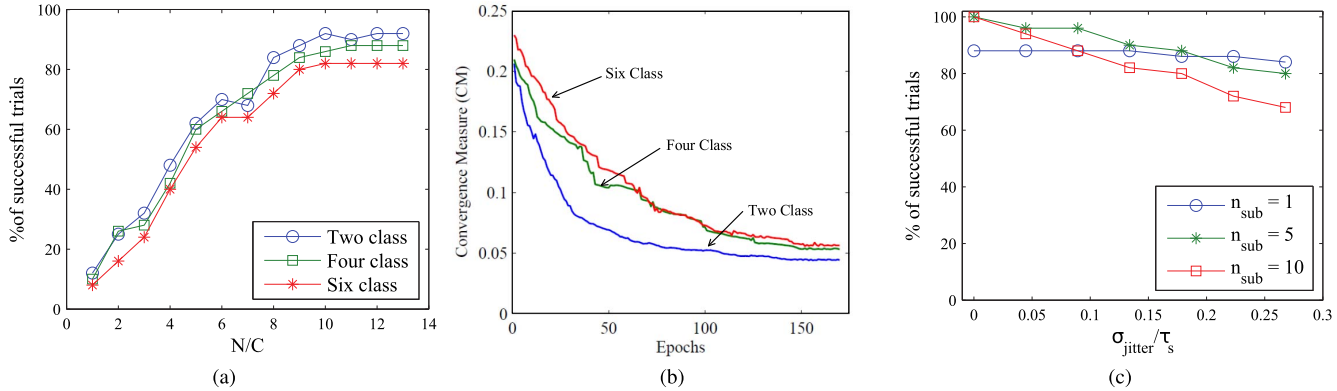

Fig. 6. (a) Percentage of successful trials is plotted against (N/C) for two-class, four-class, and six-class classification. The figure shows that as (N/C) increases, the percentage of successful trials also increases and becomes constant after $(N/C) = 11$. (b) Evolution of CM (averaged over 50 trials) with the number of epochs for two-class, four-class, and six-class classification for $n_{\text{sub}} = 1$. (c) Percentage of successful trials is plotted against $\sigma_{\text{jitter}}/\tau_s$ for $n_{\text{sub}} = 1$, 5, and 10 for $C = 4$. As the number of subpatterns is increased, the jitter/noise robustness of the network decreases.

network is obtained. If we are considering d dimensional patterns and the mean firing rate of each dimension is μ_f, then the mean ISI across the entire pattern is given by $\mu_{\text{ISI}} = 1/(d \times \mu_f)$. We can then set $\tau_{s,\text{opt}}$ according to the formula

$$\tau_{s,\text{opt}} = 52.83\mu_{\text{ISI}} - 3.1. \tag{13}$$

In our simulations, we keep τ_f as $\tau_f = (\tau_s/10)$. Since the weights of all the active synapses are 1, we set $I_0 = 1.4351$ to normalize the amplitude of the PSC to be 1.

E. Threshold of Nonlinearity (x_{thr})

During the training of WTA-NNLD, the STDP-NRW rule preferably selects those connection topologies where correlated inputs for synaptic connections are connected to the same branch. Thus, the lumped dendritic nonlinearity $b(z) = (z^2/x_{\text{thr}})$ should give a supralinear output only when correlated input dimensions are connected to the dendrite. To ensure this, we keep the value of x_{thr} equal to the average input to the nonlinear function in case of random connections. We create numerous instances of dendrites having k synapses and calculate the average input to the nonlinear function, $b_{\text{in,avg}}$, for the given pattern set. Then, we set the value of x_{thr} as $x_{\text{thr}} = b_{\text{in,avg}}$.

F. V_{thr} of NNLD

The NNLD should provide a postsynaptic spike only when correlated inputs have been connected to its dendrites. We consider an NNLD having m dendrites and k synapses and create numerous instances of random connections to these synapses. We measure the average value of the maximum membrane voltage $[(V_{\text{max}})_{\text{av}}]$ produced when this NNLD integrated spikes over the pattern duration for all these instances and set $V_{\text{thr}} = (V_{\text{max}})_{\text{av}}$.

G. Normalization Constant ($I_{0,inh}$), Slow ($\tau_{s,inh}$), and Fast ($\tau_{f,inh}$) Time Constant of $I_{inh}(t)$

The postsynaptic firing activity of the WTA-NNLD network is dependent on $\tau_{s,\text{inh}}$ and $I_{0,\text{inh}}$. To simulate the hardware

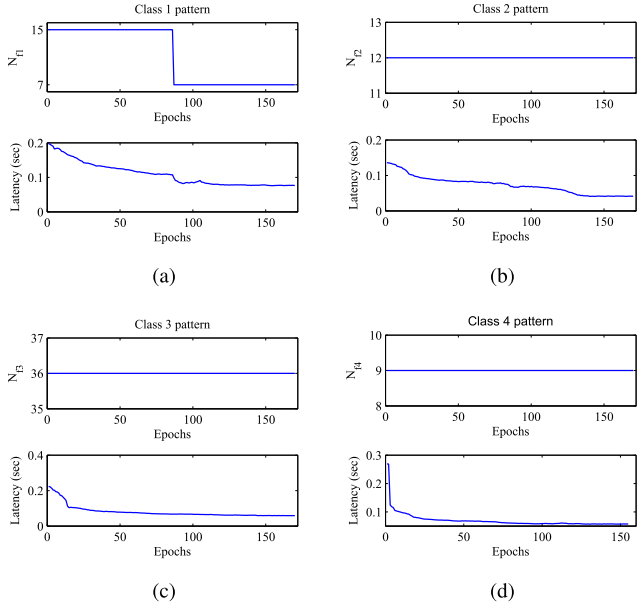

Fig. 7. For four-class classification $(C = 4)$, the above figure shows that out of 44 NNLDs (a) 7th NNLD recognizes class 1 pattern, (b) class 2 pattern fires for the 12th NNLD, (c) 36th NNLD recognizes class 3 pattern, and (d) class 4 pattern is recognized by 9th NNLD and the latency for all four of them decreases over epochs until saturation.

scenario, we set $\tau_{f,\text{inh}}$ to a small value given by $\tau_{f,\text{inh}} = (\tau_{s,\text{inh}}/10)$. To set $I_{0,\text{inh}}$ and $\tau_{s,\text{inh}}$, we first excite WTA-NNLD with ep_{ini} epochs of patterns prior to training and calculate the average excitatory current $(I_{e,\text{av}})$ to the NNLDs as

$$I_{e,\text{av}} = \left\langle \frac{1}{N} \sum_{n=1}^{N} I_{\text{in}}^{n}(t) \right\rangle_{ep_{ini}} \tag{14}$$

where $\langle\cdot\rangle_{ep_{\text{ini}}}$ denotes averaging over ep_{ini} epochs. The idea is to generate $I_{\text{inh}}(t)$, which, if provided by N_{inh} at the beginning of a subpattern, decays exponentially to $I_{e,\text{av}}$ at the end of the subpattern, i.e., after time T_{sub} has elapsed. This ensures that once a postsynaptic spike is generated by an NNLD in a particular T_{sub} time window, other NNLDs are unable to fire during that the same T_{sub} time window.

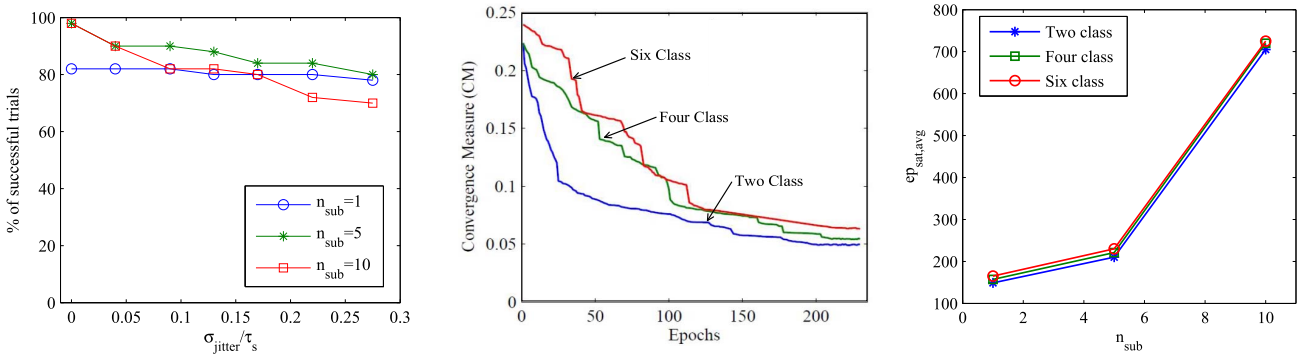

Fig. 8. (a) Percentage of successful trials for $C = 4$ is plotted against $\sigma_{\text{jitter}}/\tau_s$ for $n_{\text{sub}} = 1$, 5, and 10 for patterns having spikes in only 50% of the afferents. (b) Evolution of CM (averaged over 50 trials) with the number of epochs for two-class, four-class, and six-class classification when $n_{\text{sub}} = 5$. (c) Number of epochs needed for saturation of CM (averaged over 50 trials) against the number of subpatterns considered for each pattern (n_{sub}). As n_{sub} increases, WTA-NNLD has to train itself for more number of subpatterns, and hence, there is an increase in $ep_{\text{sat,avg}}$.

Assuming $\tau_{f,\text{inh}} \ll \tau_{s,\text{inh}}$, we can derive that the required $I_{\text{inh}}(t)$ is implemented by setting $\tau_{s,\text{inh}}$ as

$$\tau_{s,\text{inh}} = \frac{T_{\text{sub}}}{\ln\left(\frac{I_{0,\text{inh}}}{I_{e,\text{av}}}\right)}. \tag{15}$$

Note that $\tau_{s,\text{inh}}$ has an inverse logarithmic relation to $I_{0,\text{inh}}$.

H. Convergence Measure

The formula for calculating CM, applicable to both $n_{\text{sub}} = 1$ and $n_{\text{sub}} > 1$, for detecting the convergence of learning is given by

$$\text{CM} = \frac{l_{\text{mean}}}{n_{\text{sub}}} - \frac{(n_{\text{sub}} - 1)T_{\text{sub}}}{2}. \tag{16}$$

Note that for $n_{\text{sub}} = 1$, CM $= l_{\text{mean}}$ and so it computes the time-to-first spike for patterns averaged over an epoch. For $n_{\text{sub}} > 1$, CM calculates the average time-to-first spike from the beginning of each subpattern of C patterns of an epoch. We consider the learning has converged when the value of CM saturates.

IV. Experiments and Results

In this section, we will describe the classification task considered in this paper. To show how the network performs for multiclass classification, we will consider two, four, and six classes of patterns. We will be showing the performance of WTA-NNLD and STDP-NRW for three values of n_{sub} given by $n_{\text{sub}} = 1$, 5, and 10.

A. Problem Description

The benchmark task we have selected to analyze the performance of the proposed method is the spike train classification problem [31]. In the generalized spike train classification problem, C arrays of h Poisson spike trains having frequency f and length T_p are present, which are labeled as classes 1 to C. Jittered versions of these templates are created by altering the position of each spike within the templates by a random amount that is randomly drawn from a Gaussian distribution with zero mean and standard deviation σ_{jitter}. The network is trained by these jittered versions of spike trains, and the task is to correctly identify a pattern's class. In this paper, unless otherwise mentioned, we have considered $h = 100$, $f = 20$, $T_p = 0.5$ s, and varied C and σ_{jitter}. Inspired by [15], we also consider the scenario when $h/2$ randomly chosen afferents do not contain any spikes, while the remaining $h/2$ afferents have Poisson spike trains.

B. Case 1: $n_{sub} = 1$

In this case, we have $T_{\text{sub}} = T_p$, so one NNLD is capable of firing only once when a pattern is presented. Considering $\sigma_{\text{jitter}} = 0$, we have varied the number of NNLDs and noted the percentage of successful trials, as shown in Fig. 6(a). To make the horizontal axis invariant of the number of classes, we have taken (N/C) as the horizontal axis. From Fig. 6(a), we can conclude that the percentage of successful trials gradually increases with an increase in (N/C) and finally becomes constant after $(N/C) = 11$. Thus, unless otherwise mentioned, we will keep $N = 11 \times C$ when $n_{\text{sub}} = 1$. It can be seen from Fig. 6(a) that the percentage of successful trials cannot go beyond 92%, 88%, and 82% for two-class, four-class, and six-class classification, respectively.

Earlier in Section III, we mentioned that the learning converges when CM saturates. For $n_{\text{sub}} = 1$, CM $= l_{\text{mean}}$ is the average of the time-to-first spikes for patterns in an epoch. As an example, we consider a particular trial of four-class classification and show in Fig. 7 that during training, the latencies of the four NNLDs, N_{f1}, N_{f2}, N_{f3}, and N_{f4}, which uniquely recognize the four classes of patterns gradually reduce until reaching a saturation point. Moreover, in Fig. 6(b), we show the epochwise evolution of CM averaged over 50 trials for two-class, four-class, and six-class classification. It is evident from Fig. 6(b) that the value of CM decreases, thereby showing that the algorithm is favoring correlated inputs such that the postsynaptic spikes can occur faster and finally saturates after some epochs have passed. We denote the number of epochs taken by the algorithm for saturation of CM as ep_{sat} and note its value for 50 trials. The average

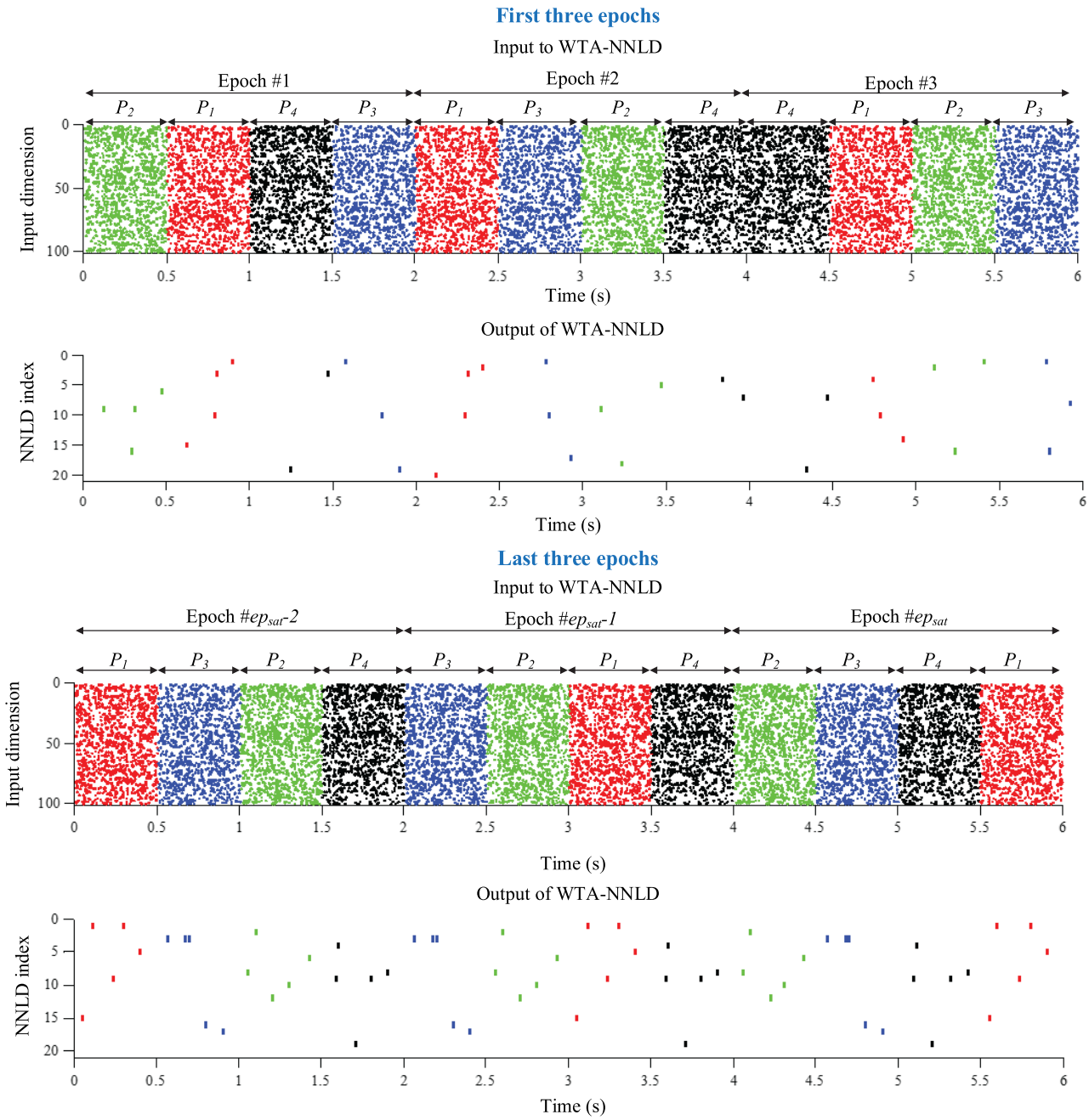

Fig. 9. Input and output of WTA-NNLD have been shown for a particular trial of the four-class classification when $n_{\text{sub}} = 5$. P_1, P_2, P_3, and P_4 represent the patterns of a particular class. The figure shows that before learning WTA-NNLD produces arbitrary spikes whenever a pattern is presented. After learning, the network produces unique sequence of NNLD spikes for patterns of different classes. This unique sequence acts as an identifier of the pattern class.

value of ep_{sat} for 50 trials, $ep_{\text{sat,avg}}$, is then computed to be $ep_{\text{sat,avg}} = 149$, 157, and 165 for two-class, four-class, and six-class classification, respectively. Moreover, this phenomenon clearly indicates that while the WTA-NNLD network is being trained by STDP-NRW learning rule, C unique NNLDs, which have locked onto the C different classes of pattern, are trying to recognize the start of repeating patterns for different classes. Fig. 6(b) also suggests that after the training has stopped, these C unique NNLDs, instead of looking at the whole pattern of duration $T_p = 500$ ms, can now look only at the starting 44.8, 53.1, and 56.1 ms of the patterns for $C = 2$, 4, and 6, respectively, to predict its class.

Let us now consider the effect of jitter and we show in Fig. 6(c) the performance of the proposed method when the intensity of jitter is varied. Next, we look into the performance of the proposed method when patterns with 50% empty afferents are considered. Fig. 8(a) shows the results obtained by our network for this case with varying amounts of jitter.

C. Case 2: $n_{sub} > 1$

Next, we consider $n_{\text{sub}} = 5$, i.e., we divide each pattern into five subpatterns by setting $\tau_{s,\text{inh}}$ and $I_{0,\text{inh}}$ as per (15). For C class classification, the maximum number of subpatterns

can be $C \times n_{\text{sub}}$ so we set $N = C \times n_{\text{sub}}$ in this case, i.e., we keep $N = 10$, 20, and 30 for two-class, four-class, and six-class classification, respectively. Considering $\sigma_{\text{jitter}} = 0$, the evolution of CM with epochs for two-class, four-class, and six-class classification averaged over 50 trials is shown in Fig. 8(b). Moreover, the value of $ep_{\text{sat,avg}}$ (averaged over 50 trials) is found out to be 210, 221, and 230 when $C = 2$, 4, and 6, respectively. Unlike Case 1, here, we consider the response to a pattern as a unique firing sequence of few NNLDs. As an example, we consider a particular trial of four-class classification and look into the first and last three epochs during its training. It is evident from Fig. 9 that during the first three epochs, WTA-NNLD produces arbitrary sequences of spikes. However, it can be seen that after the training of the network is complete, WTA-NNLD produces different firing sequences for different patterns while producing the same sequence when the same patterns are encountered. WTA-NNLD trained by this method produces 100% accuracy in recognizing different patterns by producing its unique firing sequence for $C = 2$, 4, and 6. The performance of the network with varying intensity of jitter is shown in Fig. 6(c) (spikes present in all afferents) and Fig. 8(a) (spikes present in only half of the afferents), which depict that the $n_{\text{sub}} = 5$ case is less jitter resilient than the $n_{\text{sub}} = 1$ case.

We further increase the resolution of pattern subdivision by decreasing $\tau_{s,\text{inh}}$. We consider $n_{\text{sub}} = 10$ and following the principle of $n_{\text{sub}} = 5$, the number of NNLDs employed for $n_{\text{sub}} = 10$ is 20, 40, and 60 for two-class, four-class, and six-class classification, respectively. This approach also provides 100% accuracy in providing a unique sequence of firing whenever a particular pattern is encountered when $\sigma_{\text{jitter}} = 0$. However, the performance of the network falls rapidly with the increase in σ_{jitter}, as shown in Figs. 6(c) and 8(a). We also show the evolution of CM with the epochs in Fig. 8(b). Furthermore, the number of epochs needed for convergence of CM, in this case, is much more than the previous cases, as shown in Fig. 8(c). We conclude that dividing a pattern into too many subpatterns hampers the network performance.

Next, we delve a bit further and show the statistics of causes for the failure of the system in producing successful trials. A trial may fail if either condition 1) or 2) (described in Section II-D) is not satisfied. We denote the failure of condition 1) as $F1$. Note that for a trial, condition 2) can fail if a pattern is misclassified as a pattern of another class (denoted by $F2$) or as a random pattern (denoted by $F3$). Table I shows the statistics of failed trials for $n_{\text{sub}} = 1$, 5, and 10 when $C = 4$ and $\sigma_{\text{jitter}}/\tau_s \approx 0.1$. Note that $F1$ is high for $n_{\text{sub}} = 1$, since a unique NNLD might lock onto multiple classes of patterns. $F1$ reduces for $n_{\text{sub}} = 5$ and increases again for $n_{\text{sub}} = 10$, since sometimes the network fails to produce a unique ten indices long representation for all patterns of the same class.

Moreover, we test our network with random patterns and note the cases where a learned unique representation is produced for a random input pattern, i.e., a false positive error occurs. The percentages of false positive errors produced for $n_{\text{sub}} = 1$, 5, and 10 when $\sigma_{\text{jitter}}/\tau_s \approx 0.1$ are 8%, 0%, and 0%, respectively. Note that no false positive errors occur for $n_{\text{sub}} = 5$ and 10, since it is highly unlikely for a random pattern to make a sequence of neuron firing the same as any of the learned representation.

TABLE I
ANALYSIS OF FAILURE STATISTICS

Case	$n_{sub} = 1$	$n_{sub} = 5$	$n_{sub} = 10$
$F1$	12%	2%	6%
$F2$	2%	2%	2%
$F3$	2%	4%	8%

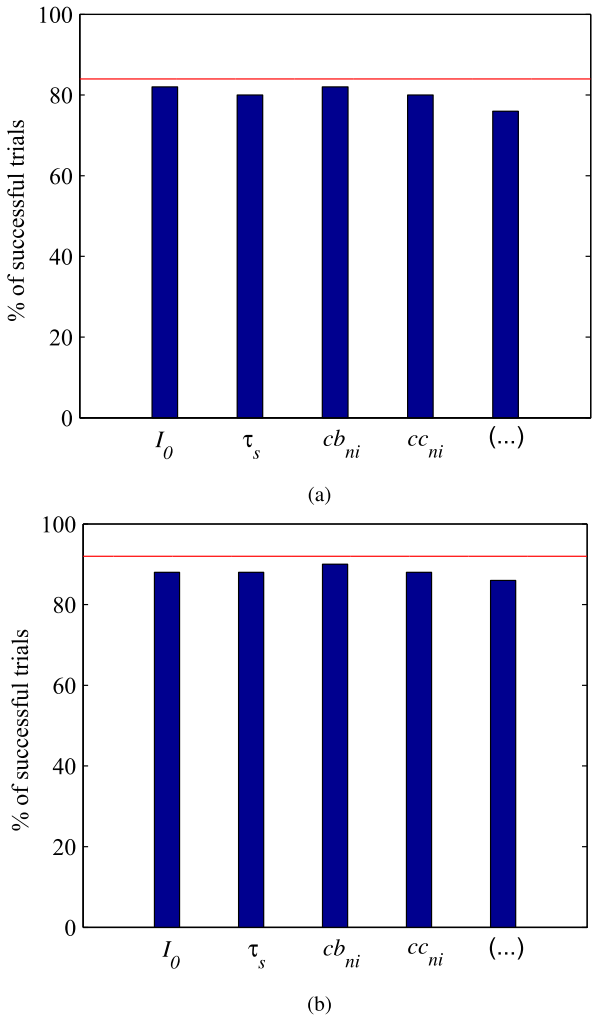

Fig. 10. Stability of WTA-NNLD trained by STDP-NRW is plotted with respect to different hardware nonidealities for (a) $n_{\text{sub}} = 1$ and (b) $n_{\text{sub}} = 5$. The constant red line indicates the percentage of successful trials obtained by our method without any nonidealities when $\sigma_{\text{jitter}}/\tau_s \approx 0.1$. The bars represent the percentage of successful trials obtained after inclusion of nonidealities. The rightmost bar marked by (...) represents the performance when all the nonidealities are included simultaneously.

V. VLSI IMPLEMENTATION: EFFECT OF STATISTICAL VARIATION

In this section, we analyze the stability of our algorithm to hardware nonidealities by incorporating the statistical variations of the key subcircuits. The primary subcircuits needed to implement our architecture are synapse, dendritic squaring block, neuron, and c^n_{pj} calculator. While the variabilities of the

synapse circuit are modeled by mismatch in the amplitude (I_0) and time constant (τ_s) of the synaptic kernel function, the variabilities of the squaring block are captured by a multiplicative constant (cb_{ni}) [29]. We do not consider the variation of the inhibitory current kernel, since it is global and only a single instance of it is present in the architecture. In our earlier work [29], [32], we proposed the circuits for implementing the synapse and squaring block of NNLD and performed Monte Carlo analysis to find their variabilities. We presented that the (σ/μ) value of I_0, τ_s, and cb_{ni} for the worst case scenario are 13%, 10.1%, and 18%, respectively. The mismatch of the LIF neuron circuit proposed in [32] was captured by variations in the firing threshold V_{thr}, the (σ/μ) value of which was computed to be 12.5%. Finally, the nonidealities of the c^n_{pj} calculator block, described in [33], are modeled as a multiplicative constant (cc_{ni}). Monte Carlo analysis of the c^n_{pj} calculator block revealed that its (σ/μ) for the worst case is 18%.

Fig. 10 shows the performance of the proposed method when these nonidealities are included in the model for $n_{\text{sub}} = 1$ and $n_{\text{sub}} = 5$ keeping $\sigma_{\text{jitter}}/\tau_s \approx 0.1$. The bars corresponding to I_0, τ_s, cb_{ni}, and cc_{ni} denote the performance degradation when statistical variations of I_0, τ_s, cb_{ni}, and cc_{ni} are included individually. The results of Fig. 10(a) and (b) show that the performance of the proposed algorithm is most affected by τ_s and cc_{ni} and least by cb_{ni}. Finally, to mimic the proper hardware scenario, we consider the simultaneous implementations of all the nonidealities, which is marked by (...). The (...) bars show that there is an 8% and 6% decrease in the performance for $n_{\text{sub}} = 1$ and $n_{\text{sub}} = 5$, respectively.

VI. Conclusion

We have proposed a new neuroinspired WTA architecture (WTA-NNLD) and an STDP inspired dendrite specific structural plasticity-based learning rule (STDP-NRW) for its training. Motivated by recent biological evidence and models suggesting nonlinear processing properties of neuronal dendrites, we employ NNLD to construct our WTA architecture. Moreover, we consider binary synapses instead of high-resolution synaptic weights. Hence, our learning rule, instead of weight updates, trains the network by modification of the connections between input and synapses. We have also provided a method by which the number of subpatterns per pattern learned by WTA-NNLD can be controlled. WTA-NNLD encodes the patterns of different classes by either activity of distinct NNLDs or by a distinct sequence of NNLD firings. To demonstrate the performance of WTA-NNLD and STDP-NRW, we have considered two-class, four-class, and six-class classification of 100-D Poisson spike trains. We can conclude from the result that the slow time constant of inhibitory signal ($\tau_{s,\text{inh}}$) can be properly set to obtain a tradeoff between specificity and sensitivity of the network. Our immediate future work will include studying the effects of connection changes after the network gets integrated over multiple patterns. This will reduce the number of required computations. On another note, we will look into the classification of spike-based MNIST [34], [35] data sets by our method. Our network can be immediately scaled to learn the digits of MNIST data set, the only requirement being additional simulation time and computational memory compared with the tasks considered in this paper. Furthermore, to achieve invariance to scaling and rotation during image classification, we will be constructing NNLD-based convolutional neural networks [36] trained by structural plasticity. We will also implement the proposed network in hardware and apply it for real time online unsupervised classification of spatio-temporal spike trains.

References

[1] M. Riesenhuber and T. Poggio, "Hierarchical models of object recognition in cortex," *Nature Neurosci.*, vol. 2, no. 11, pp. 1019–1025, Nov. 1999.

[2] W. Maass, "Neural computation with winner-take-all as the only nonlinear operation," in *Advances in Neural Information Processing Systems*, vol. 12. Cambridge, MA, USA: MIT Press, 2000, pp. 293–299.

[3] W. Maass, "On the computational power of winner-take-all," *Neural Comput.*, vol. 12, no. 11, pp. 2519–2535, Nov. 2000.

[4] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998.

[5] S. Kaski and T. Kohonen, "Winner-take-all networks for physiological models of competitive learning," *Neural Netw.*, vol. 7, nos. 6–7, pp. 973–984, 1994.

[6] J. A. Barnden and K. Srinivas, "Temporal winner-take-all networks: A time-based mechanism for fast selection in neural networks," *IEEE Trans. Neural Netw.*, vol. 4, no. 5, pp. 844–853, Sep. 1993.

[7] Y. He and E. Sanchez-Sinencio, "Min-net winner-take-all CMOS implementation," *Electron. Lett.*, vol. 29, no. 14, pp. 1237–1239, Jul. 1993.

[8] J. Starzyk and X. Fang, "CMOS current mode winner-take-all circuit with both excitatory and inhibitory feedback," *Electron. Lett.*, vol. 29, no. 10, pp. 908–910, May 1993.

[9] T. Serrano and B. Linares-Barranco, "A modular current-mode high-precision winner-take-all circuit," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 5. May 1994, pp. 557–560.

[10] G. Indiveri, "Winner-take-all networks with lateral excitation," *Analog Integr. Circuits Signal Process.*, vol. 13, nos. 1–2, pp. 185–193, 1997.

[11] G. Indiveri, "A current-mode hysteretic winner-take-all network, with excitatory and inhibitory coupling," *Anal. Integr. Circuits Signal Process.*, vol. 28, no. 3, pp. 279–291, Sep. 2001.

[12] S.-C. Liu, "A normalizing a VLSI network with controllable winner-take-all properties," *Analog Integr. Circuits Signal Process.*, vol. 31, no. 1, pp. 47–53, 2002.

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

[14] 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, 2013, doi: 10.3389/fnbot.2013.00010.

[15] T. Masquelier, R. Guyonneau, and S. J. Thorpe, "Competitive STDP-based spike pattern learning," *Neural Comput.*, vol. 21, no. 5, pp. 1259–1276, May 2009.

[16] A. Delorme, L. Perrinet, and S. J. Thorpe, "Networks of integrate-and-fire neurons using rank order coding B: Spike timing dependent plasticity and emergence of orientation selectivity," *Neurocomputing*, vols. 38–40, pp. 539–545, Jun. 2001.

[17] R. Guyonneau, R. VanRullen, and S. J. Thorpe, "Temporal codes and sparse representations: A key to understanding rapid processing in the visual system," *J. Physiol.-Paris*, vol. 98, nos. 4–6, pp. 487–497, 2004.

[18] T. Masquelier and S. J. Thorpe, "Unsupervised learning of visual features through spike timing dependent plasticity," *PLoS Comput. Biol.*, vol. 3, no. 2, p. e31, Feb. 2007, doi: 10.1371/journal.pcbi.0030031.

[19] W. Gerstner, R. Ritz, and J. L. van Hemmen, "Why spikes? Hebbian learning and retrieval of time-resolved excitation patterns," *Biol. Cybern.*, vol. 69, nos. 5–6, pp. 503–515, 1993.

[20] M. Yoshioka, "Spike-timing-dependent learning rule to encode spatiotemporal patterns in a network of spiking neurons," *Phys. Rev. E*, vol. 65, p. 011903, Dec. 2001.

[21] B. Nessler, M. Pfeiffer, L. Buesing, and W. Maass, "Bayesian computation emerges in generic cortical microcircuits through spike-timing-dependent plasticity," *PLoS Comput. Biol.*, vol. 9, no. 4, p. e1003037, 2013.

[22] D. Kappel, B. Nessler, and W. Maass, "STDP installs in winner-take-all circuits an online approximation to hidden Markov model learning," *PLoS Comput. Biol.*, vol. 10, no. 3, p. e1003511, Mar. 2014.

[23] S. Hussain, S.-C. Liu, and A. Basu, "Hardware-amenable structural learning for spike-based pattern classification using a simple model of active dendrites," *Neural Comput.*, vol. 27, no. 4, pp. 845–897, 2015.

[24] K. A. Boahen, "Point-to-point connectivity between neuromorphic chips using address events," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 47, no. 5, pp. 416–434, May 2000.

[25] S. Brink *et al.*, "A learning-enabled neuron array IC based upon transistor channel models of biological phenomena," *IEEE Trans. Biomed. Circuits Syst.*, vol. 7, no. 1, pp. 71–81, Feb. 2013.

[26] P. Poirazi and B. W. Mel, "Impact of active dendrites and structural plasticity on the memory capacity of neural tissue," *Neuron*, vol. 29, no. 3, pp. 779–796, Mar. 2001.

[27] S. Hussain, R. Gopalakrishnan, A. Basu, and S.-C. Liu, "Morphological learning: Increased memory capacity of neuromorphic systems with binary synapses exploiting AER based reconfiguration," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Aug. 2013, pp. 1–7.

[28] S. Roy, A. Basu, and S. Hussain, "Hardware efficient, neuromorphic dendritically enhanced readout for liquid state machines," in *Proc. IEEE Biomed. Circuits Syst. (BioCAS)*, Oct./Nov. 2013, pp. 302–305.

[29] S. Roy, A. Banerjee, and A. Basu, "Liquid state machine with dendritically enhanced readout for low-power, neuromorphic VLSI implementations," *IEEE Trans. Biomed. Circuits Syst.*, vol. 8, no. 5, pp. 681–695, Oct. 2014.

[30] M. P. Jadi, B. F. Behabadi, A. Poleg-Polsky, J. Schiller, and B. W. Mel, "An augmented two-layer model captures nonlinear analog spatial integration effects in pyramidal neuron dendrites," *Proc. IEEE*, vol. 102, no. 5, pp. 782–798, May 2014.

[31] T. Natschläger, H. Markram, and W. Maass, *Computer Models and Analysis Tools for Neural Microcircuits*. Boston, MA, USA: Kluwer, 2002, ch 9.

[32] A. Banerjee, S. Kar, S. Roy, A. Bhaduri, and A. Basu, "A current-mode spiking neural classifier with lumped dendritic nonlinearity," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2015, pp. 714–717.

[33] S. Roy, S. K. Kar, and A. Basu, "Architectural exploration for on-chip, online learning in spiking neural networks," in *Proc. 14th Int. Symp. Integr. Circuits (ISIC)*, Dec. 2014, pp. 128–131.

[34] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[35] T. S. Gotarredona and B. L. Barranco. *MNIST-DVS and FLASH-MNIST-DVS Databases*, accessed on Sep. 21, 2014. [Online]. Available: http://www2.imse-cnm.csic.es/caviar/MNISTDVS.html

[36] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1998, pp. 255–258.

Subhrajit Roy (S'13) was born in Kolkata, India, in 1989. He received the B.E. degree from the Department of Electronics and Telecommunication Engineering, Jadavpur University, Kolkata, India, in 2012, and the Ph.D. degree from Nanyang Technological University, Singapore, in 2016, where he is currently pursuing the Ph.D. degree in electrical engineering under the supervision of Prof. A. Basu.

He is currently a Research Fellow with the VIRTUS IC Design Centre of Excellence, School of Electrical and Electronic Engineering, Nanyang Technological University. He has authored research articles in peer-reviewed journals and international conference proceedings, including the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS, PART B, the IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS, the *Information Sciences*, the *Engineering Optimization*, the *Applied Soft Computing*, the *Progress in Electromagnetics Research (PIER) B*, the Biomedical Circuits and Systems Conference 2013, the Genetic and Evolutionary Computation Conference 2011, and the IEEE Congress on Evolutionary Computation from 2011 to 2012. He has reviewed papers for journals and conferences, including the IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS, the *Neural Networks* (Elsevier), the *Swarm and Evolutionary Computation*, the IEEE TRANSACTIONS ON SYSTEMS, MAN AND CYBERNETICS, PART C, the International Conference on Swarm, Evolutionary, Memetic Computing from 2011 to 2013, and the International Conference on VLSI Design 2015. His current research interests include neural networks, bio-inspired circuits, neuromorphic engineering, and evolutionary computation.

Arindam Basu (M'10) received the B.Tech. and M.Tech. degrees in electronics and electrical communication engineering from IIT Kharagpur, Kharagpur, India, in 2005, the M.S. degree in mathematics and the Ph.D. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2009 and 2010, respectively.

He joined Nanyang Technological University, Singapore, in 2010, as an Assistant Professor. His current research interests include bio-inspired neuromorphic circuits, non-linear dynamics in neural systems, low power analog IC design, and programmable circuits and devices. Dr. Basu received the Prime Minister of India Gold Medal in 2005 from IIT Kharagpur (awarded to the top student). He received the best student paper award at Ultrasonics symposium in 2006, best live demonstration at ISCAS 2010, and a finalist position in the best student paper contest at the IEEE International Symposium on Circuits and Systems (ISCAS) 2008. He was awarded MIT Technology Review's inaugural TR35 at Singapore award in 2012 for being among the top 12 innovators under the age of 35 in SE Asia, Australia, and New Zealand. He is currently an Associate Editor of the IEEE SENSORS JOURNAL from 2015 to 2017 and the IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS from 2016 to 2018. He is also Guest Editor of two Special Issues in the IEEE TRANSACTIONS ON BIOMEDICAL CIRCUITS AND SYSTEMS for selected papers from the IEEE ISCAS 2015 and IEEE Biomedical Circuits and Systems 2015 conferences. He is a Technical Committee Member of the several IEEE Circuits and Systems Society technical committees.