Biologically Realizable Reward-Modulated Hebbian Training for Spiking Neural Networks

Silvia Ferrari, Bhavesh Mehta, Gianluca Di Muro, Antonius M.J. VanDongen, and Craig Henriquez

***Abstract*— Spiking neural networks have been shown capable of simulating sigmoidal artificial neural networks providing promising evidence that they too are universal function approximators. Spiking neural networks offer several advantages over sigmoidal networks, because they can approximate the dynamics of biological neuronal networks, and can potentially reproduce the computational speed observed in biological brains by enabling temporal coding. On the other hand, the effectiveness of spiking neural network training algorithms is still far removed from that exhibited by backpropagating sigmoidal neural networks. This paper presents a novel algorithm based on reward-modulated spike-timing-dependent plasticity that is biologically plausible and capable of training a spiking neural network to learn the exclusive-or (XOR) computation, through rate-based coding. The results show that a spiking neural network model with twenty-three nodes is able to learn the XOR gate accurately, and performs the computation on time scales of milliseconds. Moreover, the algorithm can potentially be verified in light-sensitive neuronal networks grown *in vitro* by determining the spikes patterns that lead to the desired synaptic weights computed *in silico* when induced by blue light *in vitro*.**

I. INTRODUCTION

SIGMOIDAL neural networks are arguably one of the most powerful and flexible paradigms in the field of computational intelligence, thanks to their universal function approximation properties and their ability to learn unknown mappings via backpropagation. Recently, spiking neural networks (SNNs) have been shown capable of simulating sigmoidal artificial neural networks (ANNs) providing promising evidence that they too are universal function approximators [1]. SNNs offer several advantages over sigmoidal ANNs. Not only they are biologically plausible, but they have shown to reproduce fairly closely the dynamics of biological neuronal networks. By enabling temporal coding through spiking neurons, SNNs can potentially reproduce the computational speed observed in biological brains, which, for example, can carry out visual pattern analysis with ten-layer neuronal networks in only 100 ms [2], [3]. Moreover, SNNs can be used to build computational models of cortical regions, as shown by Henriquez in [4], because they can simulate the ability of biological neuronal networks to use the precise timing of single spikes to encode information [5], [6].

On the other hand, the effectiveness of current SNN training algorithms is still far removed from that exhibited by backpropagating sigmoidal ANNs. Consequently, one line of research has investigated modified backpropagation algorithms for SNNs (as reviewed in [7]). Although the response of certain SNN models (such as the spike response model used in this paper) is piece-wise continuous, this approach remains relatively unsuccessful due to the limited applicability of gradient descent methods to objective functions that are only continuously differentiable. More importantly, supervised backpropagation generally meets considerable sceptism in the neurobiological community due to overwhelming evidence disputing the propagation of errors in biological neuronal networks [8]. Another active line of research has investigated biologically-inspired algorithms that update the weights based on physiologically plausible mechanisms, such as, Hebbian synaptic plasticity and spike-timing-dependent synaptic plasticity (STDP), both of which are supported by considerable experimental evidence [8], [9], [10], [11], [12]. Along this line of research, this paper develops a biologically-plausible SNN training algorithm that can solve a benchmark learning problem from the ANN literature, known as the exclusive-or (XOR) gate, and is experimentally realizable *in vitro*.

One objective of this research is to develop learning rules that can be verified in light-sensitive neuronal networks grown *in vitro* by determining the spikes patterns that lead to the desired synaptic weights computed *in silico* when induced by blue light *in vitro*. Hippocampal and cortical neurons are currently being grown in dissociated culture on a multi electrode array (MEA) in the VanDongen Laboratory [13], [14], [15], [16]. This constitutes a substantial difference with the approach, described in [12], since the algorithm presented here has been designed to be experimented, through the aforementioned techniques. The use of cultured neurons enables the simultaneous recording of the firing behavior of several hundred neurons, providing data with very high spatial and temporal resolution that can be used to investigate and verify the mechanisms by which neurons store and control the flow of information in the brain. Neuronal cultures help address difficulties associated with studying the intact brain, by reducing the neuronal connectivity and diversity of cell types, and by enabling measurements from all neurons. Furthermore, they enable system-level studies about neurons reorganization and memory restoration in response to defined lesions, and coordinated activity between networks of neurons.

The authors are with Duke University, Durham, NC 27707, USA
Silvia Ferrari is Assistant Professor of Mechanical Engineering sferrari@duke.edu
Bhavesh Mehta is a graduate student of Biomedical Engineering bhavesh.mehta@duke.edu
Gianluca Di Muro is a graduate student of Mechanical Engineering gianluca.dimuro@duke.edu
Antonius M.J. VanDongen is Associate Professor of Pharmacology and Neurobiology vando005@mc.duke.edu
Craig Henriquez is Professor of Biomedical Engineering and Computer Science ch@duke.edu

978-1-4244-1821-3/08/$25.00©2008 IEEE

Since these processes involve calcium (Ca) signaling, a fluorescent Ca dye Fluo-4 can be used to visualize Ca waves for thousands of neurons, as shown in Fig. 1. Using a recent genetic advance that makes the cultured hippocampal neurons light sensitive by the expression of a light-activated cation channel, ChannelRhodopsin2 (ChR2), neurons can be made to fire a chosen number of spikes using spatiotemporal blue-light patterns that induce individual-neuron stimulation and inhibition with very high precision, and for prolonged periods of time. The effects of reward on learning and memory consolidation can also be investigated and tested *in vitro* by monitoring and stimulating the chemical activity of the neurons, including dopaminergic, adrenergic and cholinergic signaling pathways. Consequently, the training algorithm developed in this paper exploits the learning paradigm known as reward-modulated STDP [11], which adapts the synaptic efficacies according to a time-dependent reward, and the relative timing of the pre- and post-synaptic spikes. The STDP paradigm is particularly appealing because the synaptic efficacies (represented by adjustable weights in the SNN) cannot be directly manipulated *in vitro*. Instead, neurons can be made to spike at precise moments in time through spatiotemporal light patterns that can be reconstructed from the recordings obtained from the training session. The novel reward-modulated STDP training algorithm is presented in Section III, and applied to the spike response model SNN described in Section II. The results presented in Section IV demonstrate that this biologically-plausible algorithm is capable of training SNNs to learn the XOR gate computation, which is a simple and yet challenging classical benchmark problem for ANN training.

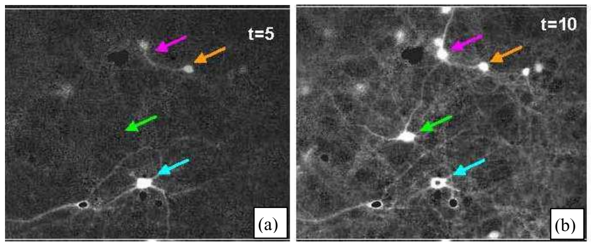

Fig. 1. Cultured hippocampal neurons grown 24 days *in vitro* and visualization of Ca waves at t = 5 sec (a) and at t = 10 sec (b), with arrows pointing at four neurons with synchronous activity.

II. Spiking Neural Network (SNN) Model

Spiking neuron models aim at providing a phenomenological model for the dynamics of biological neurons, which include action-potential generation, refractory periods, and post-synaptic potential shaping. A *spike* or action potential consists of a short electrical pulse that propagates unchanged along the axon of the firing neuron, and is delivered to other neurons. It is generated when the membrane potential of neuron i, denoted by $v_i(t)$, exceeds a threshold ϑ due to the input spikes from presynaptic neurons. Each spike has an amplitude of approximately 100 mV and lasts approximately 1 to 2 ms. However, since a neuron produces spikes that are all in the same form, it is the time sequence of spikes, or *spike train* of the neuron, that carries relevant information to other neurons. Thus, every spike is represented by the Dirac delta function $\delta(t - t_i^k)$ denoting a singularity function that occurs at time $t = t_i^k$, where i is the neuron index, and k is the spike index in the spike train.

A. Spike Response Model (SRM)

Early examples of spiking neurons were highly simplified and neglected many aspects of neuronal dynamics, such as, adaptation, bursting, inhibitory rebound, and shunting inhibition [17]. In this paper, the spike response model (SRM) is used to represent the neuronal dynamics, as it has been shown to approximate the Hogkin-Huxley model with high accuracy [18], and gives rise to networks that are trainable and mathematically tractable. Unlike earlier models involving memoryless binary neurons, the SRM neuron keeps a decaying memory of past spikes received from other neurons. Also, although it can be viewed as a generalization of the leaky integrate-and-fire model [18], the SRM expresses the membrane potential as an integral instead of a differential equation. This integral is formulated in terms of kernel response functions of the time instant at which the neuron, indexed by i, produced its last spike, denoted by $\hat{t}_i$. After firing its last spike, the evolution of the membrane potential in neuron i is approximated by,

$$\begin{aligned} u_i(t) &= \eta(t - \hat{t}_i) + \sum_j w_{ij} \sum_k \epsilon_{ij}(t - \hat{t}_i, t - t_j^k) \\ &\quad + \int_0^{\infty} \lambda(t - \hat{t}_i, \tau) I(t - \tau) d\tau \end{aligned} \tag{1}$$

where j is the index of all presynaptic neurons, and w_{ij} is the adjustable parameter or *weight* representing the efficacy of the synapse where the axon of the presynaptic neuron j makes contact with the soma of the postsynaptic neuron i. $I(\cdot)$ is an external driving current that determines the resting potential expressed by the integral in (1) and, in this paper, is set to a small random signal.

The kernel response functions $\eta(\cdot)$, $\epsilon(\cdot)$, and $\lambda(\cdot)$ represent the temporal effects of spike firing and spike reception on the membrane potential of the neuron. The kernel $\eta(\cdot)$ describes the *refractory response* due to the last spike, and captures the form of the action potential, which consists of a positive pulse, and of the subsequent spike afterpotential. The latter consists of a negative overshoot representing the hyperpolarization phase experienced by the neuron after it fires, and before it returns to its resting potential. The refractory response is modeled as,

$$\eta(x) = -\eta_0 e^{-x/\tau_f} H(x) \tag{2}$$

based on experimental studies on motoneurons [19], where η_0 is a scaling constant, τ_f is the refractory time constant, and $H(\cdot)$ is the Heaviside function. The kernel $\lambda(\cdot)$ captures the linear response of the membrane potential to an impulse current input, which decays more rapidly when the neuron has been recently active. It is expressed as a decaying exponential with time constant τ_m, times a recovery factor that scales the magnitude of the postsynaptic potential by the amount of time that has elapsed since the neuron last fired a spike:

$$\lambda(x,\tau) = \frac{R}{\tau_m}\left[1 - e^{-x/\tau_r}\right] e^{-\tau/\tau_m} H(\tau)H(x-\tau) \quad (3)$$

Where, $\tau_r >> \tau_f$ is the response recovery time constant.

The kernel $\epsilon(\cdot)$ describes the *postsynaptic potential* evoked by the reception of a spike from a presynaptic neuron. The functional form of the postsynaptic potential is obtained by mapping integrate-and-fire neurons to the SRM (as shown in [18]):

$$\begin{aligned}\epsilon(x,y) &= \int_0^x e^{-\tau/\tau_m}\alpha(t-\tau)d\tau \\ &= \frac{e^{-t/\tau_s}}{\tau_s}\int_0^x e^{-\tau(1/\tau_m - 1/\tau_s)}H(t-\tau)d\tau \quad (4)\end{aligned}$$

Where τ_m is an effective passive membrane time constant, and $\alpha(\cdot)$ is the postsynaptic current, which is assumed to decay exponentially with time constant τ_s. The capacitance and charge are set equal to one, for simplicity. As observed in biologically neurons, the shape of the above kernel allows to reproduce a neuron's refractory behavior by which its response to a presynaptic spike is considerably reduced if the neuron has recently fired. Excitatory and inhibitory postsynaptic potentials (EPSP and IPSP, respectively) are obtained depending on the sign of w_{ij}.

Unlike integrate-and-fire neuron models, the SRM is characterized by a phenomenological dynamic threshold model:

$$\vartheta(t-\hat{t}_i) = \begin{cases} \vartheta_f, & 0 < t - \hat{t}_i < T_f \\ \vartheta_0\left[1 - e^{-(t-\hat{t}_i - T_f)/\tau_\vartheta}\right], & t - \hat{t}_i \geq T_f \end{cases} \quad (5)$$

Thus, the threshold has a high constant value, e.g., ϑ_f = 100 mV, during the absolute refractory period T_f, to prevent the neuron from firing. After this period, the threshold decreases from zero to a steady-state value ϑ_0 = − 50 mV, with a time constant τ_ϑ = 6 ms, reproducing the behavior of a full cortical neuron model [18, pg. 131]. In the SRM, neuron i fires a spike when its membrane potential increases up to the threshold and exceeds it, at which point the value of the neuron's last firing time is reset, i.e.:

$$\begin{aligned} &t_i^f = t, \text{ when } u_i(t) = \vartheta(t-\hat{t}_i) \text{ and } \tfrac{du_i(t)}{dt} > 0 \\ &\hat{t}_i = \sup_{t_i^f \in S_i(t)} t_i^f \end{aligned} \quad (6)$$

Where, the set $S_i(t) = \{t_i^1, t_i^2, \ldots\}$ is the spike train of neuron i at time t, and sup denotes the supremum of $S_i(t)$.

B. SRM Population Dynamics

One approach to studying a population of SRM spiking neurons is to consider a fully-connected network, with the same synaptic efficacy among all neurons [18, pp. 204-207]. In this case, the term representing the postsynaptic potential in (1) is re-written in terms of a pre-specified average population activity and an escape noise model that replaces (5) with a noisy threshold. Subsequently, spikes occur according to a probability density function, referred to as *firing intensity*, that depends on the difference between the deterministic membrane potential (1) and a random threshold. Under these assumptions, the dynamics of the population can be analyzed by deriving a partial differential equation for the averaged refractory density, representing how many neurons are in a refractory state that falls within a specified time interval.

In this paper, a feedforward homogeneous network of N SRM spiking neurons is considered, with non-uniform synaptic weights to be determined through training (Section III). Each neuron is represented by the membrane potential (1), and is subject to stochastic effects caused, for example, by random fluctuations in neurotransmitters and thermal noise, and represented by $I(\cdot)$ in (1). In particular, a zero-mean random variable with a variance of 20 nA is used to simulate a random current injection into the cell body which, in turn, causes a baseline fluctuation in the resting potential of the cell resulting into a baseline spiking rate of 8 Hz (for zero synaptic weights).

A subset of the neurons labeled by the index set $\mathcal{I}$, referred to as *inputs*, can be made to fire desired spike trains. The response of the entire population is then simulated by solving a set of N equations in the form (1), with $i = 1, 2, \ldots, N$, over a period of time $t \in [0, T]$. Although the response of every neuron can be recorded from the simulation, only the spike trains of a subset of *output* neurons, labeled by the index set $\mathcal{O}$, are used for temporal coding, i.e., to extract information from the response of the network. The remaining neurons, labeled by the index set $\mathcal{H}$, are referred to as *hidden*, because their response is not directly controlled, nor can be used to evaluate the network's performance during training. In the next section, an algorithm is presented for determining the synaptic weights of this SRM SNN, such that its output response can be decoded and used to perform a desired task.

III. Biologically-Inspired Training Algorithm

A biologically-plausible training algorithm that is realizable in neuronal cultures grown *in vitro* is developed by accounting for local effects, such as, refractory period, synapses' eligibility, and STDP, modulated by a global time-dependent reward signal. Let w_{ij} denote the adjustable synaptic weight for the ordered pair of pre- and post-synaptic neurons (j, i) in a feedforward SRM SNN, where $j \in \mathcal{H}$ if $i \in \mathcal{O}$, and $j \in \mathcal{I}$ if $i \in \mathcal{H}$. The weight magnitude is assumed to be bounded by a positive constant w_{max} at all times, such that $-w_{max} \leq w_{ij} \leq w_{max}$ for $\forall i, j$. Suppose the target function $y = h(x)$ specifies the desired SNN response, where $y \in \mathbb{R}^m$ denotes the target output, and $x \in \mathbb{R}^n$ denotes the target input. The SNN input and output layers are designed such that to every output neuron there corresponds an output variable, $y = \{y_i\}_{i \in \mathcal{O}}$, and to every input neuron there corresponds an input variable, $x = \{x_i\}_{i \in \mathcal{I}}$. Then, training can be formulated as the problem of finding a set of weights such that the feedforward SRM SNN learns the target function.

Similarly to biological neuronal networks, SNNs code information in the spike train of each neuron with a temporal resolution that is on a millisecond time scale [20], by a process known as temporal coding. Information can be carried either by individual spike timing or by firing rates and, therefore, can be coded based on the temporal structure or the frequency of a spike train, respectively. In this paper, rate-based coding is adopted, representing each value of the input and output variables by an average value of firing rate for the corresponding neuron. Where, the average firing rate of a neuron i over a time period $[0,\ T]$ is defined as

$$\phi_i(T) \equiv \frac{1}{T}\int_0^T \sum_{t_i^k \in S_i(T)} \delta(t - t_i^k)dt \tag{7}$$

Based on rate-coding, target input spike trains are used to form a training set $\mathcal{T}$ encoding all possible values of x. Then, the SNN response to target input spike trains in $\mathcal{T}$ can be decoded to obtain the SNN estimate of the output, $\hat{y}(x)$, which can be compared to the target output, $y = h(x)$, to establish the SNN reward. An example of rate-based coding is provided in Section III.

Training is conducted by discretizing a time period $[0,\ T_{train}]$ by an interval Δt, and by updating every synaptic weight according to the rule,

$$w_{ij}(t + \Delta t) = w_{ij}(t) + \Delta w_{ij}(t) \tag{8}$$

at every time step $t \in (0,\ T_{train}]$. The weight increment in (8) is computed as the contribution of a reward function r, an STDP term f, and the synapses' eligibility g:

$$\Delta w_{ij}(t) = \mu \cdot r(t) \cdot f(\hat{t}_i, \hat{t}_j) \cdot g[w_{ij}(t)] \tag{9}$$

Where, the learning rate μ is a constant of order $O(w_{max})$. The reward function,

$$r(t) = [b(\hat{y}, y) + r(t - \Delta t)] \cdot e^{-(t-\hat{t}_i)/\tau_c} \tag{10}$$

mimics the injection of a chemical reward that decays exponentially over time with time constant τ_c, and delivers a positive reward when the decoded SNN output, $\hat{y}$, matches the target output, y, and a negative reward otherwise. Thus, $b(\hat{y}, y)$ is a binary operator that takes the value $+1$ when $\hat{y} = y$, and the value -1 when $\hat{y} \neq y$. The STDP term,

$$f(\hat{t}_i, \hat{t}_j) = \text{sgn}(\hat{t}_i - \hat{t}_j) \cdot e^{-|\hat{t}_i - \hat{t}_j|/\tau_d} \tag{11}$$

represents the phenomenon by which when the i^{th} and the j^{th} neurons fire spikes that are close in time, their synaptic efficacy undergoes a greater change [20], as illustrated in Fig. 2. Where, sgn$(\cdot)$ is the signum function, $|\cdot|$ is the absolute value, and τ_d is a reference time delay. Finally, the synapses' eligibility is modeled by the function,

$$g[w_{ij}(t)] = 1 - c_1 \cdot e^{-c_2 \cdot |w_{ij}(t)|/w_{max}} \tag{12}$$

to take into account the phenomenon by which synapses with a higher efficacy typically undergo a greater change due to learning. Where, c_1 and c_2 are two user-chosen positive constants.

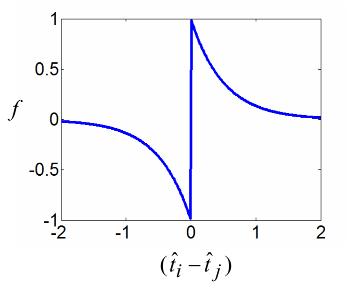

Fig. 2. STDP term as a function of the time delay between the last spike of neuron i and the last spike of neuron j.

At every time step the weights are updated starting from the output layer, proceeding to the hidden layer, and then to the input layer, using the following algorithm.

Algorithm 1 SNN Learning algorithm

```
for $t = \Delta t$ to $T_{train}$ do
  for $i = N, N-1, \ldots, 1$ do
    for every pair $(j, i)$ do
      $\Delta w_{i,j}(t) = \mu \cdot r(t) \cdot f(\hat{t}_i, \hat{t}_j) \cdot g[w_{ij}(t)]$
      $w_{i,j}(t + \Delta t) = w_{i,j}(t) + \Delta w_{i,j}(t)$
    end for
  end for
  $t = t + \Delta t$
end for
```

IV. Spiking Neural Network Training Simulations and Results

The training algorithm presented in the previous section is implemented here to train an SRM SNN to perform an exclusive-or (XOR) gate computation, a classic benchmark problem for ANN training, recently used to demonstrate learning rules for other models of SNNs. Seung demonstrated a biologically-inspired training rule through the XOR problem in integrate-by-fire SNNs [21]. Xie used the XOR problem to demonstrate a learning rule reproducing short-term facilitation and depression for SNNs implementing a Poisson escape model [22]. Florian demonstrated another type of STDP learning rule derived from Markov decision processes, involving a noisy threshold to perform the XOR computation [11]. The XOR gate is a binary operator denoted by $\veebar$ that takes $n = 2$ binary inputs, x_1 and x_2, and returns an $m = 1$ binary output $y = x_1 \veebar x_2$, such that $y = 0$ when $x_1 = x_2$, and $y = 1$ when $x_1 \neq x_2$. All variables take the values 1 or 0, to obtain a logic gate. Learning an XOR gate requires a multi-layer architecture for artificial perceptrons, and can be challenging for SNNs because the inputs $x_1 = x_2 = 1$ must be suppressed to produce a zero output.

The SNN architecture illustrated in Fig. 3, containing two input neurons, one output neuron, and twenty hidden neurons, is found capable of learning the XOR gate by means of Algorithm 1. A set comprised of target input spike trains $\mathcal{T} = \{S_1^{\ell^*}(t_{\text{train}}), S_2^{\ell^*}(t_{\text{train}})\}_{\ell=1,\dots,q}$, and a time step $\Delta t =$ 3 ms are used for training. Where, the asterisks $(*)$ denotes a target value, the superscript ℓ indexes the training sample, and the subscript $i = 1, 2$ is the index of the two input neurons coding x_1 and x_2. Each target spike train lasts for a time period t_{train} = 0.5 s, and is generated using a Poisson distribution,

$$P(n,t) = e^{-\nu t}\frac{(\nu t)^n}{n!}, \quad n = 0, 1, \dots \tag{13}$$

by letting the time interval $t = t_{\text{train}}$, and by choosing the Poisson rate ν and the number of arrivals (or spikes) n to match the target firing rate. Firing rates of 0 Hz and 40 Hz are used to code the binary values 0 and 1, respectively. Thus, $\mathcal{T}$ is obtained by generating a pair of target spike trains, $\{S_1^{\ell^*}(t_{\text{train}}), S_2^{\ell^*}(t_{\text{train}})\}$, for each combination of XOR input values, and by ordering the resulting four pairs randomly. The process is repeated $q/4$ times, finally generating a set $\mathcal{T}$ that is presented to the SNN over a training period $T_{\text{train}} = q \cdot t_{\text{train}}$. In this paper, q is varied between 20 and 400, and the larger values of q led to the best performance. Both during and after training the SNN is simulated using the neural Circuit SIMulator (CSIM) provided in [23], which implements an adaptive time-step Crank-Nicolson integrator.

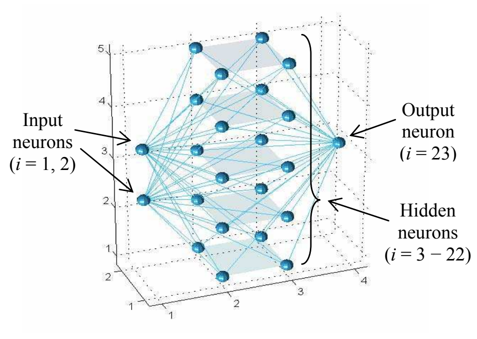

Fig. 3. Architecture of feedforward SRM spiking neural network.

Since the response of the output neurons to the input targets cannot be controlled, the reward function (10) is provided as a feedback to the network that is based on how close the decoded SNN output, $\hat{y}$, is to the target output, $y = h(x)$, for a given input $x = (x_1, x_2)$. Where, $(\hat{\cdot})$ represents the SNN estimate of the XOR gate output, $y = x_1 \veebar x_2$. The output of the SNN consists of the time history of the membrane potential of the N^{th} (output) neuron u_N, modeled as (1), which can be translated into an output spike train, $S_N(t)$. The firing rate (7) obtained from $S_N(t)$ encodes $\hat{y}$. During training, $t \in [0,\ T_{\text{train}}]$, and the output spike train is monitored to determine the reward feedback at every time step Δt. According to Algorithm 1, the network is provided with a decaying positive reward when $\hat{y}(x) = h(x) = x_1 \veebar x_2$, and with a negative decaying reward when $\hat{y}(x) \neq h(x) = x_1 \veebar x_2$. In order to monitor the SNN progress during a training session, the time history of the sum $\sum_t b(\hat{y}, y)$ is plotted in Fig. 4. The increasing trend of this sum indicates that the SNN is learning the XOR computation, and that its performance is constantly improving over the training period T_{train} = 10 sec. The spike trains of all $N = 23$ neurons during the same training session are plotted in Fig. 5.

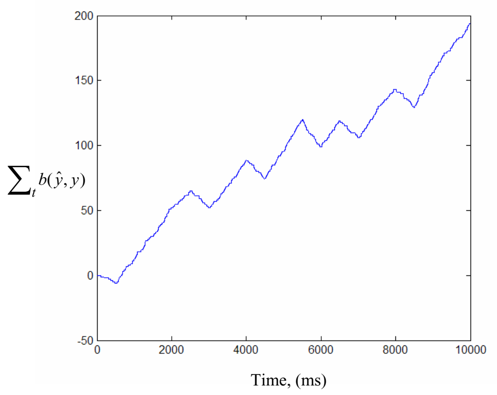

Fig. 4. The cumulative reward used for training the SNN in Fig. 3 is plotted as a function of time during XOR-gate training.

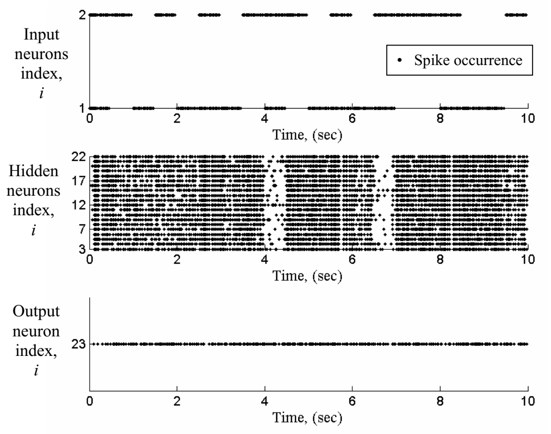

Fig. 5. Spike trains of the SNN neurons in Fig. 3 during XOR-gate training.

After training, the SNN output is decoded by assigning high firing rates of approximately 40 Hz to the output value $\hat{y} = 1$, and low firing rates of approximately 20 Hz to the output value $\hat{y} = 0$. Although similar to the rate-based criterion used for coding the SNN target input spike trains, output decoding differs from the former because output firing rates near zero cannot be achieved in the presence of a random resting potential (Section II-B). The non-zero resting potential results in a baseline spike rate (8 Hz) which represents the neurons' spontaneous tendency to fire randomly when there are no synaptic connections or incoming action potentials (i.e., weights are equal to zero). After training the weights are typically non-zero and, therefore, the spontaneous firings are propagated and amplified by the connections even when the input neurons have zero firing rate (e.g., their spikes are inhibited to code input $(x_1, x_2) = (0, 0)$). The performance of the trained SNN is tested by decoding its output in response to four pairs of target spike trains generated from (13) for each combination of XOR input values.

The trivial case $y = 0$ when $x_1 = x_2 = 0$ is always verified in practice, because in the absence of input spikes, output spikes are observed in response solely to the resting potential. The two cases with $y = 1$, namely $(x_1, x_2) = (0, 1)$ and $(1, 0)$, are properly computed by the trained SNN, as demonstrated in Figs. 6 and 7, where a high-frequency output spike train representing $\hat{y} = 1$ is obtained in response to either of the two input neurons displaying high firing rates. The last case, with $y = 0$ and $x_1 = x_2 = 1$, is the most challenging, because the SNN must suppress action potentials by the output neuron in response to two high-frequency input spike trains, containing numerous action potentials that tend to propagate throughout the network. This phenomenon is shown in Fig. 8, where the response is plotted for an untrained SNN with the same architecture (Fig. 3) and weight values that are randomly chosen between $-w_{max}$ and w_{max}. It can be seen that, as expected, the firing rate of the output neuron is very high and even surpasses that shown in Figs. 6-7. Instead, after the SNN is trained using Algorithm 1 (as shown in Figs. 4-5) its response to input spike trains representing $x_1 = x_2 = 1$ produces a low-frequency spike train corresponding to $\hat{y} = 0$, as shown in Fig. 9. Therefore, it can be concluded that, through training, the SNN has learned to perform all XOR computations.

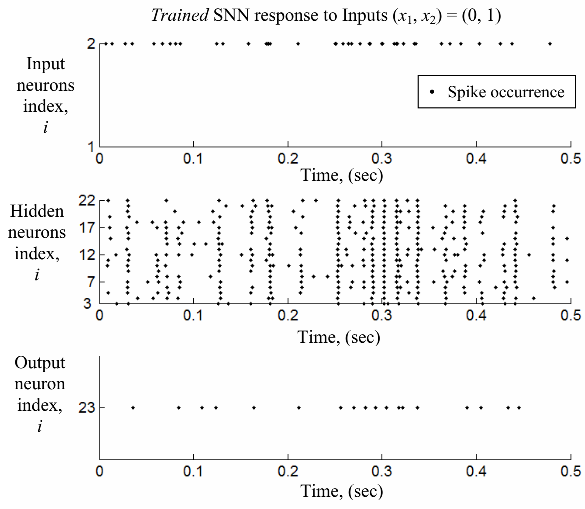

Fig. 6. Spike trains of the trained SNN response to input spike trains representing $(x_1, x_2) = (0, 1)$, obtained after XOR-gate training.

The actual firing rates of the SNN output neuron, obtained in response to input spike trains representing the four cases of the XOR-gate inputs are summarized in Table I. Firing rates are shown for a SNN with the architecture in Fig. 3 before training (first column). These firing rates can be compared to those of two networks, SNN$_1$ and SNN$_2$, with the architecture in Fig. 3, but different weights obtained through training from two different XOR training sets, that are both generated by the aforementioned procedure. It can be seen from Table I that the firing rates of the trained SNNs are very close to the target rates, for all four cases. Whereas, the untrained SNN perform very poorly in the case $(x_1, x_2) = (1, 1)$, when the frequent action potentials of the input neurons propagate through the network and cause the output neurons to also fire very frequently, thus producing the output $\hat{y} = 1$, instead of the target XOR output $y = 0$.

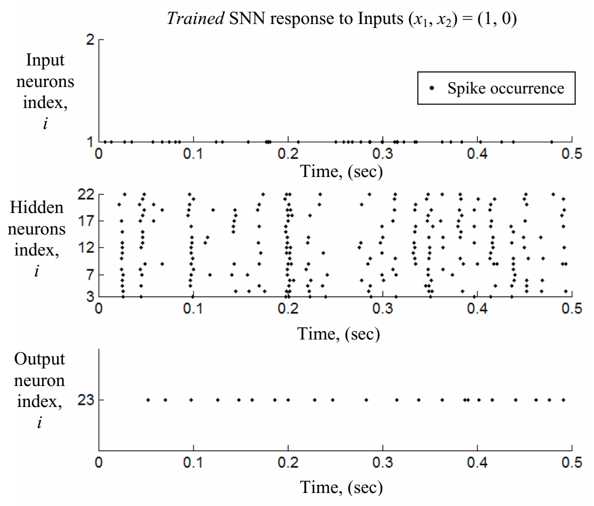

Fig. 7. Spike trains of the trained SNN response to input spike trains representing $(x_1, x_2) = (1, 0)$, obtained after XOR-gate training.

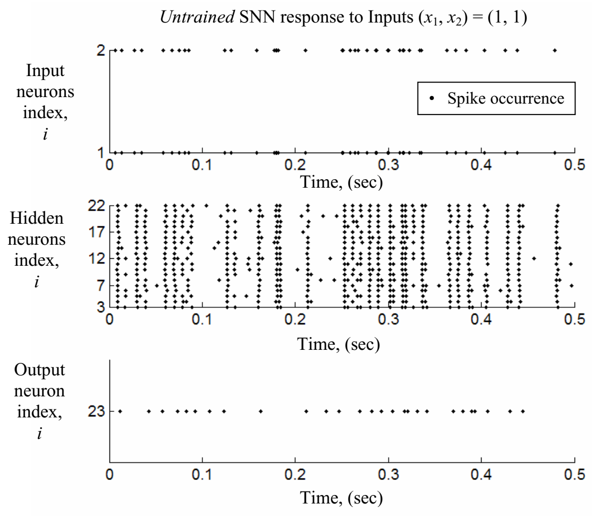

Fig. 8. Spike trains of an untrained SNN response to input spike trains representing $(x_1, x_2) = (1, 1)$, obtained before XOR-gate training.

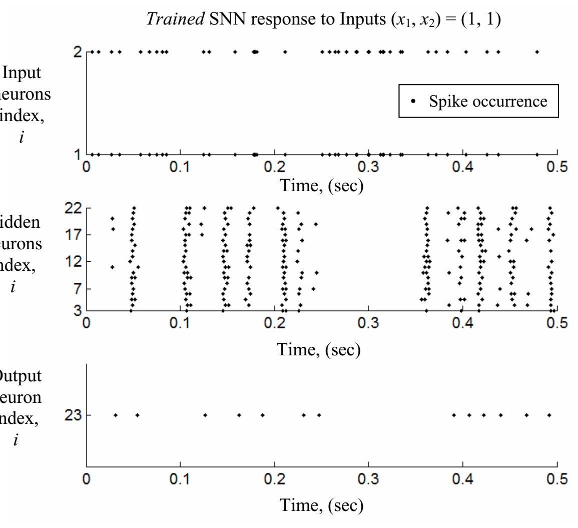

Fig. 9. Spike trains of the trained SNN response to input spike trains representing $(x_1, x_2) = (1, 1)$, obtained after XOR-gate training.

TABLE I

FIRING RATES OF SNNS BEFORE AND AFTER TRAINING THE XOR GATE.

(x_1, x_2)	Untrained SNN	Trained SNN_1	Trained SNN_2	Target Rate
(0,0)	6 (Hz)	28 (Hz)	22 (Hz)	20 (Hz)
(0,1)	36 (Hz)	38 (Hz)	44 (Hz)	40 (Hz)
(1,0)	40 (Hz)	44 (Hz)	44 (Hz)	40 (Hz)
(1,1)	54 (Hz)	26 (Hz)	28 (Hz)	20 (Hz)

V. SUMMARY AND CONCLUSIONS

This paper presents a novel SNN training algorithm that is biologically-plausible and experimentally realizable *in vitro*. The algorithm developed exploits the learning paradigm known as reward-modulated spike-timing-dependent plasticity, which adapts the synaptic efficacies according to a time-dependent reward, and the relative timing of the pre- and post-synaptic spikes. The reward function mimics the injection of a chemical reward signal that decays exponentially over time and could be tested *in vitro* by exploiting dopaminergic, adrenergic, and cholinergic signaling pathways. Spike-timing-dependent plasticity is particularly appealing because the synaptic efficacies cannot be directly manipulated *in vitro*. Instead, neurons can be made to spike at precise moments in time, through spatiotemporal light patterns that can be reconstructed from the recordings obtained during the training session. The training algorithm is demonstrated by training a spike response model neural network to learn the exclusive-or (XOR) computation, a well known benchmark problem in ANN training, that requires a multi-layer architecture for artificial perceptrons and is challenging for SNNs, because unit inputs represented by high firing rates must be suppressed to produce a zero output represented by a low firing rate. The results show that a SNN with twenty-three nodes is able to learn the XOR gate accurately, and performs the computation on time scales of milliseconds.

REFERENCES

[1] W. Maass, "Noisy spiking neurons with temporal coding have more computational power than sigmoidal neurons," *Advances in Neural Information Processing Systems*, vol. 9, 1997.

[2] D. I. Perrett, E. T. Rolls, and W. C. Caan, "Visual neurons responsive to faces in the monkey temporal cortex," *Experimental Brain Research*, vol. 47, 1982.

[3] S. J. Thorpe and M. Imbert, "Biological constraints on connectionist modelling," in *Connectionism in Perspective*, R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, and L. Steels, Eds. Elsevier, 1989.

[4] G. S. Hugh, M. Laubach, M. A. L. Nicolelis, and C. S. Henriquez, "A simulator for the analysis of neuronal ensemble activity: Application to reaching tasks," *Neurocomputing*, vol. 44, 2002.

[5] E. Salinas and T. J. Sejnowski, "Correlated neuronal activity and the flow of neural information," *Nature Reviews - Neuroscience*, vol. 2, 2001.

[6] Z. F. Mainen and T. J. Sejnowski, "Reliability of spike timing in neocortical neurons," *Science*, vol. 268, 1995.

[7] H. Burgsteiner, "Imitation learning with spiking neural networks and real-world devices," *Engineering Applications of Artificial Intelligence*, vol. 19, no. 7, 2006.

[8] C. M. A. Pennartz, "Reinforcement learning by hebbian synapses with adaptive thresholds," *Neuroscience*, vol. 81, no. 2, 1997.

[9] R. Legenstein, C. Naeger, and W. Maass, "What can a neuron learn with spike-timing-dependent plasticity?" *Neural Computation*, vol. 17, 2005.

[10] J. P. Pfister, T. Toyoizumi, D. Barber, and W. Gerstner, "Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning," *Neural Computation*, vol. 18, 2006.

[11] R. V. Florian, "Reinforcement learning through modulation of spike-timing-dependent synaptic plasticity," *Neural Computation*, vol. 19, no. 6, 2007.

[12] S. G. Wysoski, L. Benuskova, and N. Kasabov, "Adaptive learning procedure for a network of spiking neurons and visual pattern recognition," *Advanced Concepts for Intelligent Vision Systems, ACIVS, Antwerp, Lecture Notes in Computer Science*, vol. 4179, 2006.

[13] A. M. VanDongen, "Vandongen laboratory," in *http://www.vandongen-lab.com/*.

[14] J. C. Grigston, H. M. VanDongen, J. O. M. III, and A. M. VanDongen, "Translation of an integral membrane protein in distal dendrites of hippocampal neurons," *European Journal of Neuroscience*, vol. 21, 2005.

[15] T. J. V. de Ven, H. M. VanDongen, and A. M. VanDongen, "The nonkinase phorbol ester receptor alpha 1-chimerin binds the nmda receptor nr2a subunit and regulates dendritic spine density," *Journal of Neuroscience*, vol. 25, 2005.

[16] A. VanDongen, J. Codina, J. Olate, R. Mattera, R. Joho, L. Birnbaumer, and A. Brown, "Newly identified brain potassium channels gated by the guanine nucleotide binding protein go," *Science*, vol. 242, 1988.

[17] W. Maass, "Networks of spiking neurons: the third generation of neural network models," in *Proc. of the* 7^{th} *Australian Conference on Neural Networks*, Canberra, Australia, 1996.

[18] W. Gerstner and W. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*. Cambridge, UK: Cambridge University Press, 2006.

[19] R. K. D. B. Powers and M. D. Binder, "Eperimental evaluation of input-output models of motoneuron discharges," *Journal of Neurophysiology*, vol. 75, 1996.

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

[21] H. S. Seung, "Learning in spiking neural networks by reinforcement of stochastic synaptic transmission," *Neuron*, vol. 40, no. 6, 2003.

[22] X. Xie and H. S. Seung, "Learning in neural networks by reinforcement of irregular spiking," *Phisical Review E*, vol. 69, 2004.

[23] W. Maass, T. Natschläger, and H. Markram, "Computational models for generic cortical microcircuits," in *Computational Neuroscience: A Comprehensive Approach*, J. Feng, Ed. CRC-Press, 2002.