frontiers in NEUROSCIENCE

ORIGINAL RESEARCH ARTICLE
published: 30 January 2014
doi: 10.3389/fnins.2013.00272

Event-driven contrastive divergence for spiking neuromorphic systems

Emre Neftci[1]*, Srinjoy Das[1,2], Bruno Pedroni[3], Kenneth Kreutz-Delgado[1,2] and Gert Cauwenberghs[1,3]

[1] Institute for Neural Computation, University of California, San Diego, La Jolla, CA, USA
[2] Electrical and Computer Engineering Department, University of California, San Diego, La Jolla, CA, USA
[3] Department of Bioengineering, University of California, San Diego, La Jolla, CA, USA

Edited by:
André Van Schaik, The University of Western Sydney, Australia

Reviewed by:
Michael Schmuker, Freie Universität Berlin, Germany
Philip De Chazal, University of Western Sydney, Australia

***Correspondence:**
Emre Neftci, Institute for Neural Computation, University of California, San Diego, 9500 Gilman Drive - 0523, La Jolla, CA-92093, USA
e-mail: nemre@ucsd.edu

Restricted Boltzmann Machines (RBMs) and Deep Belief Networks have been demonstrated to perform efficiently in a variety of applications, such as dimensionality reduction, feature learning, and classification. Their implementation on neuromorphic hardware platforms emulating large-scale networks of spiking neurons can have significant advantages from the perspectives of scalability, power dissipation and real-time interfacing with the environment. However, the traditional RBM architecture and the commonly used training algorithm known as Contrastive Divergence (CD) are based on discrete updates and exact arithmetics which do not directly map onto a dynamical neural substrate. Here, we present an event-driven variation of CD to train a RBM constructed with Integrate & Fire (I&F) neurons, that is constrained by the limitations of existing and near future neuromorphic hardware platforms. Our strategy is based on neural sampling, which allows us to synthesize a spiking neural network that samples from a target Boltzmann distribution. The recurrent activity of the network replaces the discrete steps of the CD algorithm, while Spike Time Dependent Plasticity (STDP) carries out the weight updates in an online, asynchronous fashion. We demonstrate our approach by training an RBM composed of leaky I&F neurons with STDP synapses to learn a generative model of the MNIST hand-written digit dataset, and by testing it in recognition, generation and cue integration tasks. Our results contribute to a machine learning-driven approach for synthesizing networks of spiking neurons capable of carrying out practical, high-level functionality.

Keywords: synaptic plasticity, neuromorphic cognition, Markov chain monte carlo, recurrent neural network, generative model

1. INTRODUCTION

Machine learning algorithms based on stochastic neural network models such as RBMs and deep networks are currently the state-of-the-art in several practical tasks (Hinton and Salakhutdinov, 2006; Bengio, 2009). The training of these models requires significant computational resources, and is often carried out using power-hungry hardware such as large clusters (Le et al., 2011) or graphics processing units (Bergstra et al., 2010). Their implementation in dedicated hardware platforms can therefore be very appealing from the perspectives of power dissipation and of scalability.

Neuromorphic Very Large Scale Integration (VLSI) systems exploit the physics of the device to emulate very densely the performance of biological neurons in a real-time fashion, while dissipating very low power (Mead, 1989; Indiveri et al., 2011). The distributed structure of RBMs suggests that neuromorphic VLSI circuits and systems can become ideal candidates for such a platform. Furthermore, the communication between neuromorphic components is often mediated using asynchronous address-events (Deiss et al., 1998) enabling them to be interfaced with event-based sensors (Liu and Delbruck, 2010; Neftci et al., 2013; O'Connor et al., 2013) for embedded applications, and to be implemented in a very scalable fashion (Silver et al., 2007; Joshi et al., 2010; Schemmel et al., 2010).

Currently, RBMs and the algorithms used to train them are designed to operate efficiently on digital processors, using batch, discrete-time, iterative updates based on exact arithmetic calculations. However, unlike digital processors, neuromorphic systems compute through the continuous-time dynamics of their components, which are typically Integrate & Fire (I&F) neurons (Indiveri et al., 2011), rendering the transfer of such algorithms on such platforms a non-trivial task. We propose here a method to construct RBMs using I&F neuron models and to train them using an online, event-driven adaptation of the Contrastive Divergence (CD) algorithm.

We take inspiration from computational neuroscience to identify an efficient neural mechanism for sampling from the underlying probability distribution of the RBM. Neuroscientists argue that brains deal with uncertainty in their environments by encoding and combining probabilities optimally (Doya et al., 2006), and that such computations are at the core of cognitive function (Griffiths et al., 2010). While many mechanistic theories of how the brain might achieve this exist, a recent *neural sampling* theory postulates that the spiking activity of the neurons encodes samples of an underlying probability distribution (Fiser et al., 2010). The advantage for a neural substrate in using such a strategy over the alternative one, in which neurons encode probabilities, is that it requires exponentially fewer neurons. Furthermore, abstract model neurons consistent with the behavior of biological neurons can implement Markov Chain Monte Carlo (MCMC) sampling (Buesing et al., 2011), and RBMs sampled in this way can be efficiently trained using CD, with almost no loss in performance (Pedroni et al., 2013). We identify the conditions under which a dynamical system consisting of I&F neurons performs neural sampling. These conditions are compatible with neuromorphic implementations of I&F neurons (Indiveri et al., 2011), suggesting that they can achieve similar performance. The calibration procedure necessary for configuring the parameters of the spiking neural network is based on firing rate measurements, and so is easy to realize in software and in hardware platforms.

In standard CD, weight updates are computed on the basis of alternating, feed-forward propagation of activities (Hinton, 2002). In a neuromorphic implementation, this translates to reprogramming the network connections and resetting its state variables at every step of the training. As a consequence, it requires two distinct dynamical systems: one for normal operation (i.e., testing), the other for training, which is highly impractical. To overcome this problem, we train the neural RBMs using an online adaptation of CD. We exploit the recurrent structure of the network to mimic the discrete "construction" and "reconstruction" steps of CD in a spike-driven fashion, and Spike Time Dependent Plasticity (STDP) to carry out the weight updates. Each sample (spike) of each random variable (neuron) causes synaptic weights to be updated. We show that, over longer periods, these microscopic updates behave like a macroscopic CD weight update. Compared to standard CD, no additional connectivity programming overhead is required during the training steps, and both testing and training take place in the same dynamical system.

Because RBMs are generative models, they can act simultaneously as classifiers, content-addressable memories, and carry out probabilistic inference. We demonstrate these features in a MNIST hand-written digit task (LeCun et al., 1998), using an RBM network consisting of one layer of 824 "visible" neurons and one layer of 500 "hidden" neurons. The spiking neural network was able to learn a generative model capable of recognition performances with accuracies up to 91.9%, which is close to the performance obtained using standard CD and Gibbs sampling, 93.6%.

2. MATERIALS AND METHODS

2.1. NEURAL SAMPLING WITH NOISY I&F NEURONS

We describe here conditions under which a dynamical system composed of I&F neurons can perform neural sampling. It has been proven that abstract neuron models consistent with the behavior of biological spiking neurons can perform MCMC sampling of a Boltzmann distribution (Buesing et al., 2011). Two conditions are sufficient for this. First, the instantaneous firing rate of the neuron verifies:

$$\rho(u(t), t - t') = \begin{cases} 0 & \text{if } t - t' < \tau_r \\ r(u(t)) & t - t' \geq \tau_r \end{cases}, \tag{1}$$

with $r(u(t))$ proportional to $\exp(u(t))$, where $u(t)$ is the membrane potential and τ_r is an absolute refractory period during which the neuron cannot fire. $\rho(u(t), t - t')$ describes the neuron's instantaneous firing rate as a function of $u(t)$ at time t, given that the last spike occurred at t'. It can be shown that the average firing rate of this neuron model for stationary $u(t)$ is the sigmoid function:

$$\rho(u) = (\tau_r + \exp(-u))^{-1}. \tag{2}$$

Second, the membrane potential of neuron i is equal to the linear sum of its inputs:

$$u_i(t) = b_i + \sum_{j=1}^{N} w_{ij} z_j(t), \forall i = 1, \ldots, N, \tag{3}$$

where b_i is a constant bias, and $z_j(t)$ represents the pre-synaptic spike train produced by neuron j defined as being equal to 1 when the pre-synaptic neuron spikes for a duration τ_r, and equal to zero otherwise. The terms $w_{ij}z_j(t)$ are identified with the time course of the Post–Synaptic Potential (PSP), i.e., the response of the membrane potential to a pre-synaptic spike. The two conditions above define a neuron model, to which we refer as the "abstract neuron model." Assuming the network states are binary vectors $[z_1, \ldots, z_k]$, it can be shown that, after an initial transient, the sequence of network states can be interpreted as MCMC samples of the Boltzmann distribution:

$$\begin{aligned} p(z_1, \ldots, z_k) &= \frac{1}{Z} \exp\big(- E(z_1, \ldots, z_k)\big), \text{ with} \\ E(z_1, \ldots, z_k) &= -\frac{1}{2} \sum_{ij} W_{ij} z_i z_j - \sum_i b_i z_i, \end{aligned} \tag{4}$$

where $Z = \sum_{z_1,\ldots,z_k} \exp\big(- E(z_1, \ldots, z_k)\big)$ is a constant such that p sums up to unity, and $E(z_1, \ldots, z_k)$ can be interpreted as an energy function (Haykin, 1998).

An important fact of the abstract neuron model is that, according to the dynamics of $z_j(t)$, the PSPs are "rectangular" and non-additive since no two presynaptic spikes can occur faster than the refractive period. The implementation of synapses producing such PSPs on a large scale is very difficult to realize in hardware, when compared to first-order linear filters that result in "alpha"-shaped PSPs (Destexhe et al., 1998; Bartolozzi and Indiveri, 2007). This is because, in the latter model, the synaptic dynamics are linear, such that a single hardware synapse can be used to generate the same current that would be generated by an arbitrary number of synapses (see also next section). As a consequence, we will use alpha-shaped PSPs instead of rectangular PSPs in our models. The use of the alpha PSP over the rectangular PSP is the major source of degradation in sampling performance, as we will discuss in section 2.2.

2.1.1. Stochastic I&F neurons

A neuron whose instantaneous firing rate is consistent with Equation (1) can perform neural sampling. Equation (1) is a generalization of the Poisson process to the case when the firing probability depends on the time of the last spike (i.e., it is a renewal process), and so can be verified only if the neuron fires stochastically (Cox, 1962). Stochasticity in I&F neurons can be obtained through several mechanisms, such as a noisy reset potential, noisy firing threshold, or noise injection (Plesser and Gerstner, 2000). The first two mechanisms necessitate stochasticity in the neuron's parameters, and therefore may require specialized circuitry. But noise injection in the form of background Poisson spike trains requires only synapse circuits, which are present in many neuromorphic VLSI implementation of spiking neurons (Bartolozzi and Indiveri, 2007; Indiveri et al., 2011). Furthermore, Poisson spike trains can be generated self-consistently in balanced excitatory-inhibitory networks (van Vreeswijk and Sompolinsky, 1996), or using finite-size effects and neural mismatch (Amit and Brunel, 1997).

We show that the abstract neuron model in Equation (1) can be realized in a simple dynamical system consisting of leaky I&F neurons with noisy currents. The neuron's membrane potential below firing threshold θ is governed by the following differential equation:

$$C\frac{\mathrm{d}}{\mathrm{d}t}u_i = -g_L u_i + I_i(t) + \sigma\xi(t), \quad u_i(t) \in (-\infty, \theta), \tag{5}$$

where C is a membrane capacitance, u_i is the membrane potential of neuron i, g_L is a leak conductance, $\sigma\xi(t)$ is a white noise term of amplitude σ (which can for example be generated by background activity), $I_i(t)$ its synaptic current and θ is the neuron's firing threshold. When the membrane potential reaches θ, an action potential is elicited. After a spike is generated, the membrane potential is clamped to the reset potential u_{rst} for a refractory period τ_r.

In the case of the neural RBM, the currents $I_i(t)$ depend on the layer the neuron is situated in. For a neuron i in layer υ

$$\begin{aligned} I_i(t) &= I_i^d(t) + I_i^\upsilon(t), \\ \tau_{\mathrm{syn}}\frac{\mathrm{d}}{\mathrm{d}t}I_i^\upsilon &= -I_i^\upsilon + \sum_{j=1}^{N_h} q_{h_{ji}} h_j(t) + q_{b_i} b_{\upsilon_i}(t), \end{aligned} \tag{6}$$

where $I_i^d(t)$ is a current representing the data (i.e., the external input), I^υ is the feedback from the hidden layer activity and the bias, and the q's are the respective synaptic weights, and $b_\upsilon(t)$ is a Poisson spike train implementing the bias. Spike trains are represented by a sum of Dirac delta pulses centered on the respective spike times:

$$b_{\upsilon_i}(t) = \sum_{k \in Sp_i} \delta(t - t_k), \quad h_j(t) = \sum_{k \in Sp_j} \delta(t - t_k) \tag{7}$$

where Sp_i and Sp_j are the set of the spike times of the bias neuron b_{υ_i} and the hidden neuron h_j, respectively, and $\delta(t) = 1$ if $t = 0$ and 0 otherwise.

For a neuron j in layer h,

$$\begin{aligned} I_j(t) &= I_j^h(t), \\ \tau_{\mathrm{syn}}\frac{\mathrm{d}}{\mathrm{d}t}I_j^h &= -I_j^h + \sum_{i=1}^{N_\upsilon} q_{\upsilon_{ij}} \upsilon_i(t) + q_{b_j} b_{h_j}(t), \end{aligned} \tag{8}$$

where I^h is the feedback from the visible layer, and $\upsilon(t)$ and $b_h(t)$ are Poisson spike trains of the visible neurons and the bias neurons, defined similarly as in Equation (7). The dynamics of I^h and I^υ correspond to a first-order linear filter, so each incoming spike results in PSPs that rise and decay exponentially (i.e., alpha-PSP) (Gerstner and Kistler, 2002).

Can this neuron verify the conditions required for neural sampling? The membrane potential is already assumed to be equal to the sum of the PSPs as required by neural sampling. So to answer the above question we only need to verify whether Equation (1) holds. Equation (5) is a Langevin equation which can be analyzed using the Fokker–Planck equation (Gardiner, 2012). The solution to this equation provides the neuron's input/output response, i.e., its transfer curve (for a review, see Renart et al., 2003):

$$\rho(u_0) = \left(\tau_r + \tau_m\sqrt{\pi}\int_{\frac{u_{\mathrm{rst}}-u_0}{\sigma_V}}^{\frac{\theta-u_0}{\sigma_V}} \mathrm{d}x \exp(x^2)(1 + \mathrm{erf}(x))\right)^{-1}, \tag{9}$$

where erf is the error function (the integral of the normal distribution), $u_0 = \frac{I}{g_L}$ is the stationary value of the membrane potential when injected with a constant current I, $\tau_m = \frac{C}{g_L}$ is the membrane time constant, u_{rst} is the reset voltage, and $\sigma_V^2(u) = \sigma^2/(g_L C)$.

According to Equation (2), the condition for neural sampling requires that the average firing rate of the neuron to be the sigmoid function. Although the transfer curve of the noisy I&F neuron Equation (9) is not identical to the sigmoid function, it was previously shown that with an appropriate choice of parameters, the shape of this curve can be very similar to it (Merolla et al., 2010). We observe that, for a given refractory period τ_r, the smaller the ratio $\frac{\theta - u_{\mathrm{rst}}}{\sigma_V}$ in Equation (5), the better the transfer curve resembles a sigmoid function (**Figure 1**). With a small $\frac{\theta - u_{\mathrm{rst}}}{\sigma_V}$, the transfer function of a neuron can be fitted to

$$\nu(I) = \frac{1}{\tau_r}\left(1 + \frac{\exp(-I\beta)}{\gamma\tau_r}\right)^{-1}, \tag{10}$$

where β and γ are the parameters to be fitted. The choice of the neuron model described in Equation (5) is not critical for neural sampling: A relationship that is qualitatively similar to Equation (9) holds for neurons with a rigid (reflective) lower boundary (Fusi and Mattia, 1999) which is common in VLSI neurons, and for I&F neurons with conductance-based synapses (Petrovici et al., 2013).

This result also shows that synaptic weights q_{υ_i}, q_{h_j}, which have the units of charge are related to the RBM weights W_{ij} by a factor β^{-1}. To relate the neural activity to the Boltzmann distribution, Equation (4), each neuron is associated to a binary random variable which is assumed to take the value 1 for a duration τ_r after the

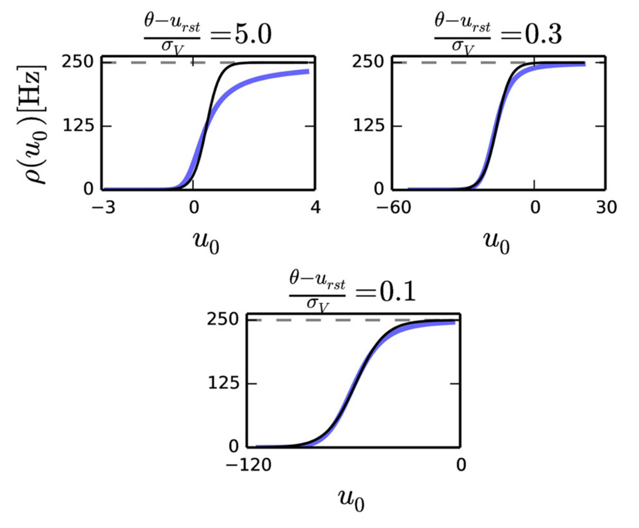


FIGURE 1 | Transfer curve of a leaky I&F neuron for three different parameter sets where $u_0 = \frac{I}{g_L}$, and $\frac{1}{\tau_r}$ = 250 [Hz] (dashed gray). In this plot, σ_V is varied to produce different ratios $\frac{\theta - u_{\mathrm{rst}}}{\sigma_V}$. The three plots above shows that the fit with the sigmoid function (solid black) improves as the ratio decreases.

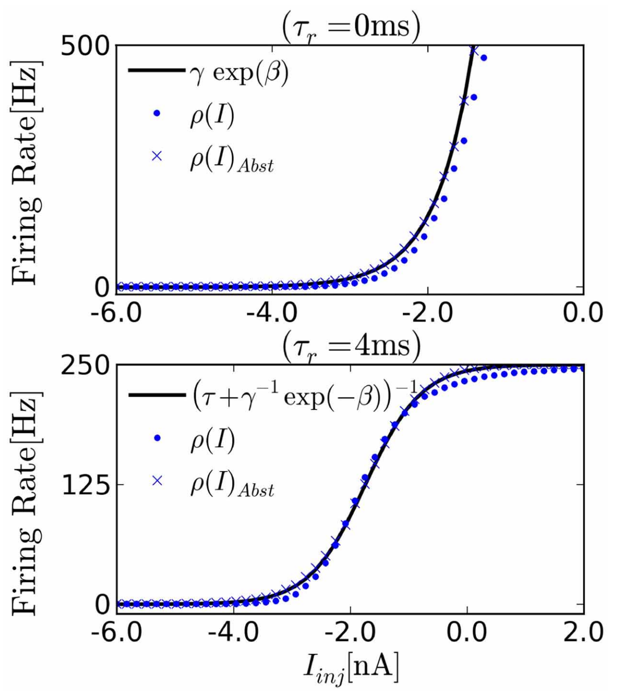


FIGURE 2 | Transfer function of I&F neurons driven by background white noise Equation (5). We measure the firing rate of the neuron as a function of a constant current injection to estimate $\rho(u_0)$, where for constant I_{inj}, $u_0 = I_{\mathrm{inj}}/g_L$. (Top) The transfer function of noisy I&F neurons in the absence of refractory period [$\rho(u) = r(u)$, circles]. We observe that ρ is approximately exponential over a wide range of inputs, and therefore compatible with neural sampling. Crosses show the transfer curve of neurons implementing the abstract neuron Equation (1), exactly. (Bottom) With an absolute refractory period the transfer function approximates the sigmoid function. The firing rate saturates at [250]Hz due to the refractory period chosen for the neuron.

neuron has spiked, and zero otherwise, similarly to Buesing et al. (2011). With this encoding, the network state is characterized by a binary vector having the same number of entries as the number of neurons in the network. The relationship between this random vector and the I&F neurons' spiking activity is illustrated in **Figure 3**. The membrane potential of the neuron (black) evolves in a random fashion until it spikes, after which it is clamped to u_{rst} for a duration τ_r (gray). While the neuron is in the refractory period, the random variable associated to it is assumed to takes the value 1. This way, the state of the network can always be associated with a binary vector. According to the theory, the dynamics in the network guarantees that the binary vectors are samples drawn from a Boltzmann distribution.

2.1.2. Calibration protocol

In order to transfer the parameters from the probability distribution Equation (4) to those of the I&F neurons, the parameters γ, β in Equation (10) need to be fitted. An estimate of a neuron's transfer function can be obtained by computing its spike rate when injected with different values of constant inputs I. The refractory period τ_r is the inverse of the maximum firing rate of the neuron, so it can be easily measured by measuring the spike rate for very high input current I. Once τ_r is known, the parameter estimation can be cast into a simple linear regression problem by fitting $\log(\rho(i)^{-1} - \tau_r)$ with $\beta I + \log(\gamma)$. **Figure 2** shows the transfer curve when $\tau_r = 0$ ms, which is approximately exponential in agreement with Equation (1).

The shape of the transfer curse is strongly dependent on the noise amplitude. In the absence of noise, the transfer curve is a sharp threshold function, which softens as the amplitude of the noise is increased (**Figure 1**). As a result, both parameters γ and β are dependent on the variance of the input currents from other neurons $I(t)$. Since $\beta q = w$, the effect of the fluctuations on the network is similar to scaling the synaptic weights and the biases which can be problematic. However, by selecting a large enough noise amplitude σ and a slow enough input synapse time constant, the fluctuations due to the background input are much larger than the fluctuations due to the inputs. In this case, β and γ remain approximately constant during the sampling.

Neural mismatch can cause β and γ to differ from neuron to neuron. From Equation (10) and the linearity of the postsynaptic currents $I(t)$ in the weights, it is clear that this type of mismatch can be compensated by scaling the synaptic weights and biases accordingly. The calibration of the parameters γ and β quantitatively relate the spiking neural network's parameters to the RBM. In practice, this calibration step is only necessary for mapping pre-trained parameters of the RBM onto the spiking neural network.

Although we estimated the parameters of software simulated I&F neurons, parameter estimation based on firing rate measurements were shown to be an accurate and reliable method for VLSI I&F neurons as well (Neftci et al., 2012).

2.2. VALIDATION OF NEURAL SAMPLING USING I&F NEURONS

The I&F neuron verifies Equation (1) only approximately, and the PSP model is different from the one of Equation (3). Therefore, the following two important questions naturally arise: how accurately does the I&F neuron-based sampler outlined above sample from a target Boltzmann distribution? How well does it perform in comparison to an exact sampler, such as the Gibbs sampler? To answer these questions we sample from several neural RBM consisting of five visible and five hidden units for randomly drawn weight and bias parameters. At these small dimensions, the probabilities associated to all possible values of the random vector **z** can be computed exactly. These probabilities are then compared to those obtained through the histogram constructed with the sampled events. To construct this histogram, each spike was extended to form a box of length τ_r (as illustrated in **Figure 3**), the spiking activity was sampled at 1 kHz, and the occurrences of all the possible 2^{10} states of the random vector **z** were counted. We added 1 to the number of occurrences of each state to avoid zero probabilities. The histogram obtained from a representative run is shown in **Figure 4** (left).

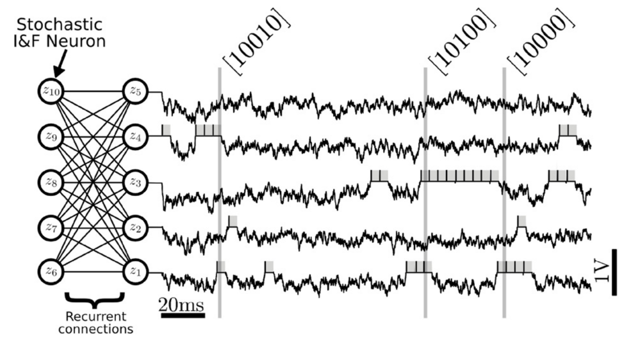


FIGURE 3 | Neural Sampling in an RBM consisting of 10 stochastic I&F neurons, with five neurons in each layer. Each neuron is associated to a binary random variable which take values 1 during a refractory period τ_r after the neuron has spiked (gray shadings). The variables are sampled at 1 kHz to produce binary vectors that correspond to samples of the joint distribution $p(z)$. In this figure, only the membrane potential and the samples produced by the first five neurons are shown. The vectors inside the brackets are example samples of the marginalized distribution $p(z_1, z_2, z_3, z_4, z_5)$ produced at the time indicated by the vertical lines. In the RBM, there are no recurrent connections within a layer.

A common measure of similarity between two distributions p and q is the KL divergence:

$$D(p||q) = \sum_i p_i \log \frac{p_i}{q_i}.$$

If the distributions p and q are identical then $D(p||q) = 0$, otherwise $D(p||q) > 0$. The right panel of **Figure 4** shows $D(p||P_{\text{exact}})$ as a function of sampling duration, for distributions p obtained from three different samplers: the abstract neuron based sampler with alpha PSPs ($P_{\text{NS,Abstract}}$), the I&F neuron-based sampler (P_{NS}), and the Gibbs sampler (P_{Gibbs}).

In the case of the I&F neuron-based sampler, the average KL divergence for 48 randomly drawn distributions after 1000 s of sampling time was 0.059 ± 0.049. This result is not significantly different if the abstract neuron model Equation (1) with alpha PSPs is used (average KL divergence 0.10 ± 0.049), and in both cases the KL divergence did not tend to zero as the number of samples increased. The only difference in the latter neuron model compared to the abstract neuron model of Buesing et al. (2011), which tends to zero when sampling time tends to infinity, is the PSP model. This indicates that the discrepancy is largely due to the use of alpha-PSPs, rather than the approximation of Equation (1) with I&F neurons.

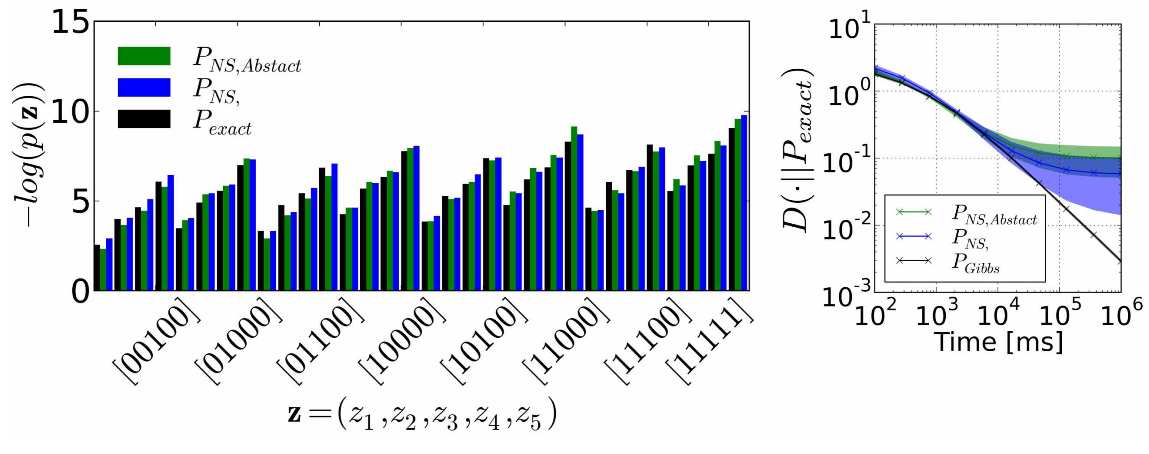


FIGURE 4 | (Left) Example probability distribution obtained by neural sampling of the RBM of **Figure 3**. The bars are marginal probabilities computed by counting the events [00000], [00001], ..., [11110], [11111], respectively. P_{NS} is the distribution obtained by neural sampling and P is the exact probability distribution computed with Equation (4). **(Right)** The degree to which the sampled distribution resembles the target distribution is quantified by the KL divergence measured across 48 different distributions, and the shadings correspond to its standard deviation. This plot also shows the KL divergence of the target distribution sampled by Gibbs Sampling (P_{Gibbs}), which is the common choice for RBMs. For comparison with the neural sampler, we identified the duration of one Gibbs sampling iteration with one refractory period $\tau_r = 4$ ms. The plot shows that up to 10^4ms, the two methods are comparable. After this, the KL divergence of the neural sampler tends to a plateau due to the fact that neural sampling with our I&F neural network is approximate. In both figures, $P_{\text{NS, Abstract}}$ refers to the marginal probability distribution obtained by using the abstract neuron model Equation (1). In this case, the KL divergence is not significantly different from the one obtained with the I&F neuron model-based sampler.

The standard sampling procedure used in RBMs is Gibbs Sampling: the neurons in the visible layer are sampled simultaneously given the activities of the hidden neurons, then the hidden neurons are sampled given the activities of the visible neurons. This procedure is iterated a number of times. For comparison with the neural sampler, the duration of one Gibbs sampling iteration is identified with one refractory period $\tau_r = 4\,\text{ms}$. At this scale, we observe that the speed of convergence of the neural sampler is similar to that of the Gibbs sampler up to 10^4ms, after which the neural sampler plateaus above the $D(p||q) = 10^{-2}$ line. Despite the approximations in the neuron model and the synapse model, these results show that in RBMs of this size, the neural sampler consisting of I&F neurons sample from a distribution that has the same KL divergence as the distribution obtained after 10^4 iterations of Gibbs sampling, which is more than the typical number of iterations used for MNIST hand-written digit tasks in the literature (Hinton et al., 2006).

2.3. NEURAL ARCHITECTURE FOR LEARNING A MODEL OF MNIST HAND-WRITTEN DIGITS

We test the performance of the neural RBM in a digit recognition task. We use the MNIST database, whose data samples consist of centered, gray-scale, 28 × 28-pixel images of hand-written digits 0–9 (LeCun et al., 1998). The neural RBM's network architecture consisted of two layers, as illustrated in **Figure 5**. The visible layer was partitioned into 784 sensory neurons (v_d) and 40 class label neurons (v_c) for supervised learning. The pixel values of the digits were discretized to two values, with low intensity pixel values ($p \leq 0.5$) mapped to 10^{-5} and high intensity values ($p > 0.5$) mapped to 0.98. A neuron i in d stimulated each neuron i in layer v, with synaptic currents f_i such that $P(\upsilon_i = 1) = \nu(f_i)\tau_r = p_i$, where $0 \leq p_i \leq 1$ is the value of pixel i. The value f_i is calculated by inverting the transfer function of the neuron: $f_i = \nu^{-1}(s) = \log\left(\frac{s}{\gamma - s\gamma\tau_r}\right)\beta^{-1}$. Using this RBM, classification is performed by choosing the most likely label given the input, under the learned model. This equals to choosing the population of class neurons associated to the same label that has the highest population firing rate.

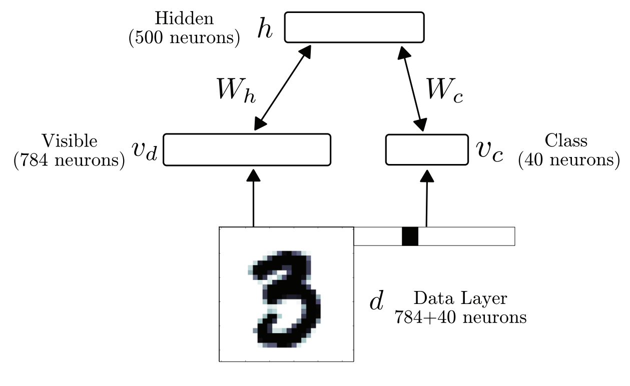


FIGURE 5 | The RBM network consists of a visible and a hidden layer. The visible layer is partitioned into 784 sensory neurons (v_d) and 40 class label neurons (v_c) for supervised learning. During data presentation, the activities in the visible layer are driven by a data layer d, consisting of a digit and its label (1 neuron per label). In the RBM, the weight matrix between the visible layer and the hidden layer is symmetric.

To reconstruct a digit from a class label, the class neurons belonging to a given digit are clamped to a high firing rate. For testing the discrimination performance of an energy-based model such as the RBM, it is common to compute the free-energy $F(\mathbf{v_c})$ of the class units (Haykin, 1998), defined as:

$$\exp(-F(v_\text{c})) = \sum_{v_\text{d},h} \exp(-E(v_\text{d}, v_\text{c}, h)), \qquad (11)$$

Table 1 | List of parameters used in the software simulations.[a]

ν_bias	Mean firing rate of bias Poisson spike train	All figures	1000 Hz
σ	Noise amplitude	All figures, except **Figure 1**	$3 \cdot 10^{-11}$ A/s$^{0.5}$
		Figure 1 (left)	$2 \cdot 10^{-11}$ A/s$^{0.5}$
		Figure 1 (right)	$3 \cdot 10^{-10}$ A/s$^{0.5}$
		Figure 1 (bottom)	$1 \cdot 10^{-9}$ A/s$^{0.5}$
β	Exponential factor (fit)	All figures	$2.044 \cdot 10^{9}$ A^{-1}
γ	Baseline firing rate (fit)	All figures	8808 Hz
τ_r	Refractory period	All figures	4 ms
τ_syn	Time constant of recurrent, and bias synapses	All figures	4 ms
τ_br	"Burn-in" time of the neural sampling	All figures	10 ms
g_L	Leak conductance	All figures	1 nS
u_rst	Reset potential	All figures	0 V
C	Membrane capacitance	All figures	10^{-12} F
θ	Firing threshold	All figures	100 mV
W	RBM weight matrix ($\in \mathbb{R}^{N_\upsilon \times N_h}$)	**Figure 4**	$N(-0.75, 1.5)$
b_υ, b_h	RBM bias for layer υ and h	**Figure 4**	$N(-1.5, 0.5)$
N_υ, N_h	Number of visible and hidden units	**Figure 4**	5, 5
	in the RBM	**Figures 7**, **8**, **7**	824, 500
N_c	Number of class label units	**Figures 7**, **8**, **7**	40
$2T$	Epoch duration	**Figures 4**, **7**, **8**	100 ms
		Figure 9	300 ms
T_sim	Simulation time	**Figure 2**	5 s
		Figure 4	1000 s
		Figure 7	0.2 s
		Figure 9	0.85 s
		Figure 8 (testing)	1.0 s
		Figure 8 (learning)	2000 s
τ_STDP	Learning time window	**Figure 7**	4 ms
η	Learning rate	Standard CD	$0.1 \cdot 10^{-2}$
		Event-driven CD	$3.2 \cdot 10^{-2}$

[a] *Software simulation scripts are available online (https://github.com/eneftci/eCD).*

and selecting v_c such that the free-energy is minimized. The spiking neural network is simulated using the BRIAN simulator (Goodman and Brette, 2008). All the parameters used in the simulations are provided in **Table 1**.

3. RESULTS

3.1. EVENT-DRIVEN CONTRASTIVE DIVERGENCE

A Restricted Boltzmann Machine (RBM) is a stochastic neural network consisting of two symmetrically interconnected layers composed of neuron-like units—a set of visible units v and a set of hidden units h, but has no connections within a layer.

The training of RBMs commonly proceeds in two phases. At first the states of the visible units are clamped to a given vector from the training set, then the states of the hidden units are sampled. In a second "reconstruction" phase, the network is allowed to run freely. Using the statistics collected during sampling, the weights are updated in a way that they maximize the likelihood of the data (Hinton, 2002). Collecting equilibrium statistics over the data distribution in the reconstruction phase is often computationally prohibitive. The CD algorithm has been proposed to mitigate this (Hinton, 2002; Hinton and Salakhutdinov, 2006): the reconstruction of the visible units' activity is achieved by sampling them conditioned on the values of the hidden units (**Figure 6**). This procedure can be repeated k times (the rule is then called CD$_k$), but relatively good convergence is obtained for the equilibrium distribution even for one iteration. The CD learning rule is summarized as follows:

$$\Delta w_{ij} = \epsilon(\langle \upsilon_i h_j \rangle_{\text{data}} - \langle \upsilon_i h_j \rangle_{\text{recon}}), \tag{12}$$

where υ_i and h_j are the activities in the visible and hidden layers, respectively. This rule can be interpreted as a difference of Hebbian and anti-Hebbian learning rules between the visible and hidden neurons sampled in the data and reconstruction phases. In practice, when the data set is very large, weight updates are calculated using a subset of data samples, or "mini-batches." The above rule can then be interpreted as a stochastic gradient descent (Robbins and Monro, 1951). Although the convergence properties of the CD rule are the subject of continuing investigation, extensive software simulations show that the rule often converges to very good solutions (Hinton, 2002).

The main result of this paper is an online variation of the CD rule for implementation in neuromorphic hardware. By virtue of neural sampling the spikes generated from the visible and hidden units can be used to compute the statistics of the probability distributions online (further details on neural sampling in the Materials and Methods section 2.1). Therefore a possible neural mechanism for implementing CD is to use synapses whose weights are governed by synaptic plasticity. Because the spikes cause the weight to update in an online, and asynchronous fashion, we refer to this rule as *event-driven* CD.

The weight update in event-driven CD is a modulated, pair-based STDP rule:

$$\frac{\mathrm{d}}{\mathrm{d}t} q_{ij} = g(t)\, \text{STDP}_{ij}(\upsilon_i(t), h_j(t)) \tag{13}$$

where $g(t) \in \mathbb{R}$ is a zero-mean global gating signal controlling the data vs. reconstruction phase, q_{ij} is the weight of the synapse and $\upsilon_i(t)$ and $h_j(t)$ refer to the spike trains of neurons υ_i and h_j, defined as in Equation (7).

As opposed to the standard CD rule, weights are updated after every occurrence of a pre-synaptic and post-synaptic event. While this online approach slightly differentiates it from standard CD, it is integral to a spiking neuromorphic framework where the data samples and weight updates cannot be stored. The weight update is governed by a symmetric STDP rule with a symmetric temporal window $K(t) = K(-t), \forall t$:

$$\begin{aligned} \text{STDP}_{ij}(\upsilon_i(t), h_j(t)) &= \upsilon_i(t) A_{h_j}(t) + h_j(t) A_{\upsilon_i}(t), \\ A_{h_j}(t) &= A \int_{-\infty}^{t} \mathrm{d}s K(t-s) h_j(s), \\ A_{\upsilon_i}(t) &= A \int_{-\infty}^{t} \mathrm{d}s K(s-t) \upsilon_i(s), \end{aligned} \tag{14}$$

with $A > 0$ defining the magnitude of the weight updates. In our implementation, updates are additive and weights can change polarity.

3.1.1. *Pairwise STDP with a global modulatory signal approximates CD*

The modulatory signal $g(t)$ switches the behavior of the synapse from LTP to LTD (i.e., Hebbian to Anti-Hebbian). The temporal average of $g(t)$ must vanish to balance LTP and LTD, and must

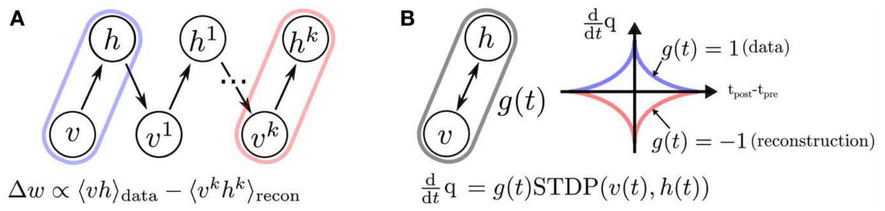

A
B
h
h^1
h^k
v
v^1
v^k
$\Delta w \propto \langle vh \rangle_{\text{data}} - \langle v^k h^k \rangle_{\text{recon}}$
$g(t)$
$\frac{d}{dt}q$
$g(t) = 1$ (data)
t_{post}-t_{pre}
$g(t) = -1$ (reconstruction)
$\frac{d}{dt}q = g(t)\text{STDP}(v(t), h(t))$

FIGURE 6 | The standard Contrastive Divergence (CD)$_k$ procedure, compared to event-driven CD. (A) In standard CD, learning proceeds iteratively by sampling in "construction" and "reconstruction" phases (Hinton, 2002), which is impractical in a continuous-time dynamical system. **(B)** We propose a spiking neural sampling architecture that folds these updates on a continuous time dimension through the recurrent activity of the network. The synaptic weight update follows a STDP rule modulated by a zero mean signal $g(t)$. This signal switches the behavior of the synapse from Long-Term Potentiation (LTP) to Long-Term Depression (LTD), and partitions the training into two phases analogous to those of the original CD rule. The spikes cause microscopic weight modifications, which on average behave as the macroscopic CD weight update. For this reason, the learning rule is referred to as event-driven CD.

vary on much slower time scales than the typical times scale of the network dynamics, denoted τ_{br}, so that the network samples from its stationary distribution when the weights are updated. The time constant τ_{br} corresponds to a "burn-in" time of MCMC sampling and depends on the overall network dynamics and cannot be computed in the general case. However, it is reasonable to assume τ_{br} to be in the order of a few refractory periods of the neurons (Buesing et al., 2011). In this work, we used the following modulation function $g(t)$:

$$g(t) = \begin{cases} 1 & \text{if } \operatorname{mod}(t, 2T) \in (\tau_{br}, T) \\ -1 & \text{if } \operatorname{mod}(t, 2T) \in (T + \tau_{br}, 2T) \\ 0 & \text{otherwise} \end{cases}, \qquad (15)$$

where mod is the modulo function and T is a time interval. The data is presented during the time intervals $(2iT, (2i+1)T)$, where i is a positive integer. With the $g(t)$ defined above, no weight update is undertaken during a fixed period τ_{br}. This allows us to neglect the transients after the stimulus is turned on and off (respectively in the beginning of the data and reconstruction phases). In this case and under further assumptions discussed below, the event-driven CD rule can be directly compared with standard CD as we now demonstrate. The average weight update during $(0, 2T)$ is:

$$\begin{aligned} \left\langle \frac{\mathrm{d}}{\mathrm{d}t} q_{ij} \right\rangle_{(0,2T)} &= C_{ij} + R_{ij}, \\ C_{ij} &= \frac{T - \tau_{br}}{2T} \left(\langle \upsilon_i(t) A_{h_j}(t) \rangle_{t_d} + \langle h_j(t) A_{\upsilon_i}(t) \rangle_{t_d} \right) \\ R_{ij} &= -\frac{T - \tau_{br}}{2T} \left(\langle \upsilon_i(t) A_{h_j}(t) \rangle_{t_r} + \langle h_j(t) A_{\upsilon_i}(t) \rangle_{t_r} \right), \end{aligned} \qquad (16)$$

where $t_d = (\tau_{br}, T)$ and $t_r = (T + \tau_{br}, 2T)$ denote the intervals during the positive and negative phases of $g(t)$, and $\langle \cdot \rangle_{(a,b)} = \frac{1}{b-a} \int_a^b \mathrm{d}t \cdot$.

We write the first average in C_{ij} as follows:

$$\begin{aligned} \langle \upsilon_i(t) A_{h_j}(t) \rangle_{t_d} &= A \frac{1}{T - \tau_{br}} \int_{\tau_{br}}^{T} \mathrm{d}t \int_{-\infty}^{t} \mathrm{d}s K(t-s) \upsilon_i(t) h_j(s), \\ &= A \frac{1}{T - \tau_{br}} \int_{\tau_{br}}^{T} \mathrm{d}t \int_{0}^{\infty} \mathrm{d}\Delta K(\Delta) \upsilon_i(t) h_j(t - \Delta), \\ &= A \int_0^\infty \mathrm{d}\Delta K(\Delta) \langle \upsilon_i(t) h_j(t-\Delta) \rangle_{t_d}. \end{aligned} \qquad (17)$$

If the spike times are uncorrelated the temporal averages become a product of the average firing rates of a pair of visible and hidden neurons (Gerstner and Kistler, 2002):

$$\langle \upsilon_i(t) h_j(t-\Delta) \rangle_{t_d} = \langle \upsilon_i(t) \rangle_{t_d} \langle h_j(t-\Delta) \rangle_{t_d} =: \bar{\upsilon}_i^+ \bar{h}_j^+.$$

If we choose a temporal window that is much smaller than T, and assume the network activity is stationary in the interval (τ_{br}, T), we can write (up to a negligible error Kempter et al., 2001)

$$\langle \upsilon_i(t) A_{h_j}(t) \rangle_{t_d} = A \bar{\upsilon}_i^+ \bar{h}_j^+ \int_0^\infty \mathrm{d}\Delta K(\Delta). \qquad (18)$$

In the uncorrelated case, the second term in C_{ij} contributes the same amount, leading to:

$$C_{ij} = \eta \bar{\upsilon}_i^+ \bar{h}_j^+.$$

with $\eta := 2A \frac{T - \tau_{br}}{2T} \int_0^\infty \mathrm{d}\Delta K(\Delta)$. Similar arguments apply to the averages in the time interval t_r:

$$R_{ij} = 2A \int_0^\infty \mathrm{d}\Delta K(\Delta) \langle \upsilon_i(t) h_j(t-\Delta) \rangle_{t_r} = \eta \bar{\upsilon}_i^- \bar{h}_j^-.$$

with $\bar{\upsilon}_i^- \bar{h}_j^- := \langle \upsilon_i(t) \rangle_{t_r} \langle h_j(t-\Delta) \rangle_{t_r}$. The average update in $(0, 2T)$ then becomes:

$$\left\langle \frac{\mathrm{d}}{\mathrm{d}t} q_{ij} \right\rangle_{(0,2T)} = \eta \left(\bar{\upsilon}_i^+ \bar{h}_j^+ - \bar{\upsilon}_i^- \bar{h}_j^- \right). \qquad (19)$$

According to Equation (18), any symmetric temporal window that is much shorter than T can be used. For simplicity, we choose an exponential temporal window $K(\Delta) = \exp(-|\Delta/\tau_{STDP}|)$ with decay rate $\tau_{STDP} \ll T$ (**Figure 6B**). In this case, $\eta = 2A \frac{T - \tau_{br}}{2T} \tau_{STDP}$.

The modulatory function $g(t)$ partitions the training into epochs of duration $2T$. Each epoch consists of a LTP phase during which the data is presented (construction), followed by a free-running LTD phase (reconstruction). The weights are updated asynchronously during the time interval in which the neural sampling proceeds, and Equation (19) tells us that its average resembles Equation (12). However, it is different in two ways: the averages are taken over one data and reconstruction phase rather than a mini-batch of data samples and their reconstructions; and more importantly, the synaptic weights are updated during the data and the reconstruction phase, whereas in the CD rule, updates are carried out at the end of the reconstruction phase. In the derivation above the effect of the weight change on the network during an epoch $2T$ was neglected for mathematical simplicity. In the following, we verify that despite this approximation, the event-driven CD performs nearly as well as standard CD in the context of a common benchmark task.

3.2. LEARNING A GENERATIVE MODEL OF HAND-WRITTEN DIGITS

We train the RBM to learn a generative model of the MNIST handwritten digits using event-driven CD (see section 2.3 for details). For training, 20,000 digits selected randomly (with repetition) from a training set consisting of 10,000 digits were presented in sequence, with an equal number of samples for each digit.

The raster plots in **Figure 7** show the spiking activity of each layer before and after learning for epochs of duration 100 ms. The top panel shows the population-averaged weight. After training, the sum of the upwards and downward excursions of the average

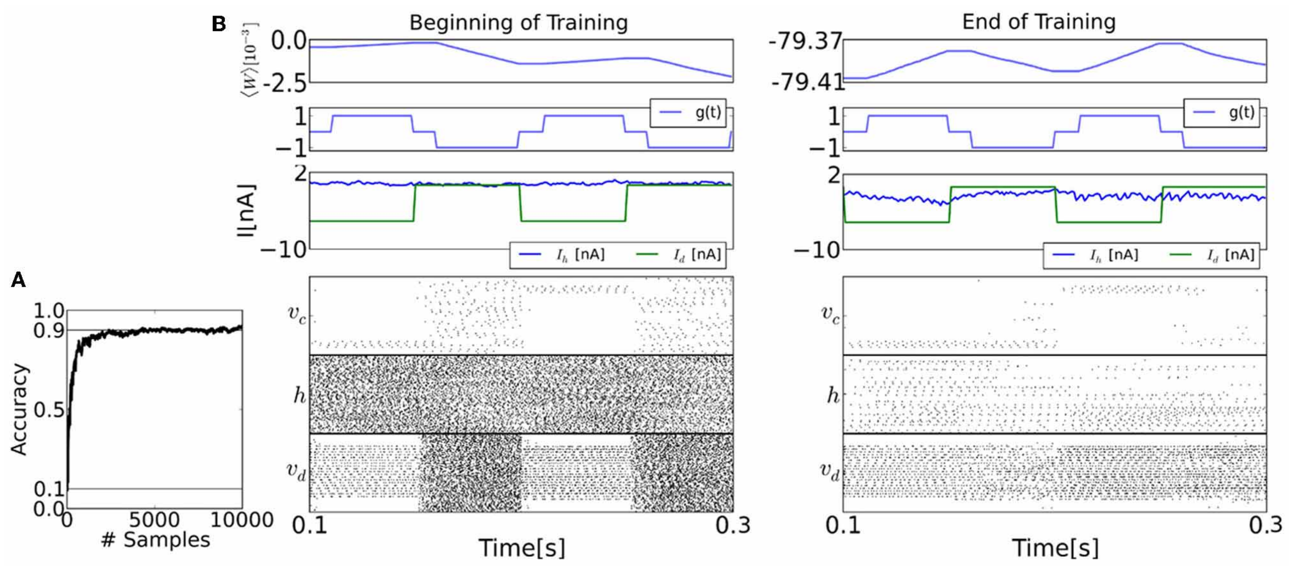


FIGURE 7 | The spiking neural network learns a generative model of the MNIST dataset using the event-driven CD procedure. (A) Learning curve, shown here up to 10, 000 samples. **(B)** Details of the training procedure, before and after training (20,000 samples). During the first half of each 0.1 s epoch, the visible layer v is driven by the sensory layer, and the gating variable g is 1, meaning that the synapses undergo LTP. During the second half of each epoch, the sensory stimulus is removed, and g is set to -1, so the synapses undergo LTD. The top panels of both figures show the mean of the entries of the weight matrix. The second panel shows the values of the modulatory signal $g(t)$. The third panel shows the synaptic currents of a visible neuron, where I_h is caused by the feedback from the hidden and the bias, and I_d is the data. The timing of the clamping (I_d) and g differ due to an interval τ_{br} where no weight update is undertaken to avoid the transients (see section 2). Before learning and during the reconstruction phase, the activity of the visible layer is random. But as learning progresses, the activity in the visible layer reflects the presented data in the reconstruction phase. This is very well visible in the layer class label neurons v_c, whose activity persists after the sensory stimulus is removed. Although the firing rates of the hidden layer neurons before training is high (average 113 Hz), this is only a reflection of the initial conditions for the recurrent couplings W. In fact, at the end of the training, the firing rates in both layers becomes much sparser (average 9.31 Hz).

weight is much smaller than before training, because the learning is near convergence. The second panel shows the value of the modulatory signal $g(t)$. The third panel shows the input current (I_d) and the current caused by the recurrent couplings (I_h).

Two methods can be used to estimate the overall recognition accuracy of the neural RBM. The first is to sample: the visible layer is clamped to the digit only (i.e., v_d), and the network is run for 1s. The known label is then compared with the position of the group of class neurons that fired at the highest rate. The second method is to minimize free-energy: the neural RBMs parameters are extracted, and for each data sample, the class label with the lowest free-energy (see section 2) is compared with the known label. In both cases, recognition was tested for 1000 data samples that were not used during the training. The results are summarized in **Figure 8**.

As a reference we provide the best performance achieved using the standard CD and one unit per class label ($N_c = 10$) (**Figure 8**, table row 1), 93.6%. By mapping the these parameters to the neural sampler, the recognition accuracy reached 92.6%. The discrepancy is expected since the neural sampler does not exactly sample from the target Boltzmann distribution (see section 2.2).

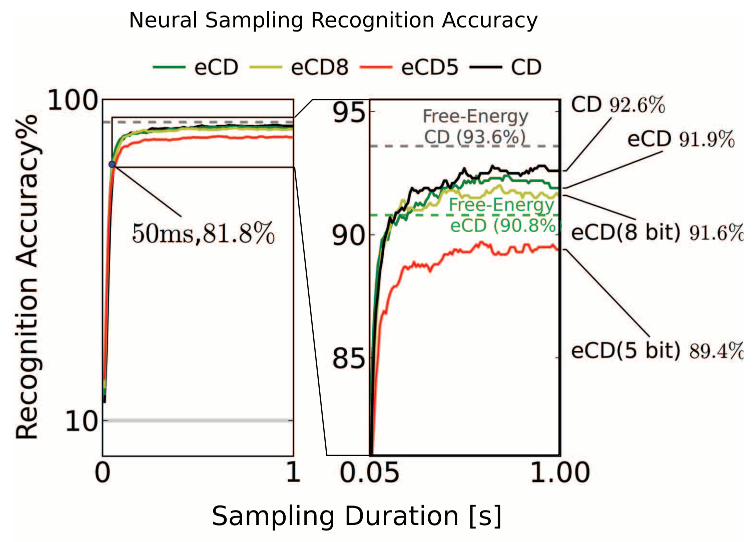
When training a neural RBM of I&F neurons using event-driven CD, the recognition result was 91.9% (**Figure 8**, table row 2). The performance of this RBM obtained by minimizing its free-energy was 90.8%. The learned parameters performed well for classification using the free-energy calculation which suggests that the network learned a model that is consistent with the mathematical description of the RBM.

In an energy-based model like the RBM the free-energy minimization should give the upper bound on the discrimination performance (Haykin, 1998). For this reason, the fact that the recognition accuracy is higher when sampling as opposed to using the free-energy method may appear puzzling. However, this is possible because the neural RBM does not exactly sample from the Boltzmann distribution, as explained in section 2.2. This suggests that event-driven CD compensates for the discrepancy between the distribution sampled by the neural RBM and the Boltzmann distribution, by learning a model that is tailored to the spiking neural network.

Excessively long training durations can be impractical for real-time neuromorphic systems. Fortunately, the learning using event-driven CD is fast: Compared to the off-line RBM training (250, 000 presentations, in mini-batches of 100 samples) the event-driven CD training succeeded with a smaller number of data presentations (20, 000), which corresponded to 2000 s of simulated time. This suggests that the training durations are achievable for real-time neuromorphic systems.

3.2.1. *The choice of the number of class neurons N_c*

Event-driven CD underperformed in the case of 1 neuron per class label ($N_c = 10$), which is the common choice for standard CD and Gibbs sampling. This is because a single neuron firing



	Accuracy Neural Sampler	Accuracy Free-energy
Standard CD	92.6%	93.6%
Event-driven CD	91.9%	90.8%
Event-driven CD (8 bits)	91.6%	91.0%
Event-driven CD (5 bits)	89.4%	89.2%

FIGURE 8 | To test recognition accuracy, the trained RBMs are sampled using the I&F neuron-based sampler for up to 1 s. The classification is read out by identifying the group of class label neurons that had the highest activity. This experiment is run for RBM parameter sets obtained by standard CD (black, CD) and event-driven CD (green, eCD). To test for robustness to finite precision weights, the RBM was run with parameters obtained by event-driven CD discretized to 8 and 5 bits. In all scenarios, the accuracy after 50 ms of sampling was above 80% and after 1 s the accuracies typically reached their peak at around 92%. The dashed horizontal lines show the recognition accuracy obtained by minimizing the free-energy (see text). The fact that the eCD curve (solid green) surpasses its free-energy line suggests that a model that is tailored to the I&F spiking neural network was learned.

at its maximum rate of 250 Hz cannot efficiently drive the rest of the network without tending to induce spike-to-spike correlations (e.g., synchrony), which is incompatible with the assumptions made for sampling with I&F neurons and event-driven CD. As a consequence, the generative properties of the neural RBM degrade. This problem is avoided by using several neurons per class label (in our case four neurons per class label) because the synaptic weight can be much lower to achieve the same effect, resulting in smaller spike-to-spike correlations.

3.2.2. Neural parameters with finite precision

In hardware systems, the parameters related to the weights and biases cannot be set with floating-point precision, as can be done in a digital computer. In current neuromorphic implementations the synaptic weights can be configured at precisions of about 8 bits (Yu et al., 2012). We characterize the impact of finite-precision synaptic weights on performance by discretizing the weight and bias parameters to 8 bits and 5 bits. The set of possible weights were spaced uniformly in the interval $(\mu - 4.5\sigma, \mu + 4.5\sigma)$, where μ, σ are the mean and the standard deviation of the parameters across the network, respectively. The classification performance of MNIST digits degraded gracefully. In the 8 bit case, it degrades only slightly to 91.6%, but in the case of 5 bits, it degrades more substantially to 89.4%. In both cases, the RBM still retains its discriminative power, which is encouraging for implementation in hardware neuromorphic systems.

3.3. GENERATIVE PROPERTIES OF THE RBM

We test the neural RBM as a generative model of the MNIST dataset of handwritten digits, using parameters obtained by running the event-driven CD. The RBM's generative property enables it to classify and generate digits, as well as to infer digits by combining partial evidence. These features are clearly illustrated in the following experiment (**Figure 9**). First the digit 3 is presented (i.e., layer υ_d is driven by layer d) and the correct class label in υ_c activated. Second, the neurons associated to class label 5 are clamped, and the network generated its learned version of the digit. Third, the right-half part of a digit 8 is presented, and the class neurons are stimulated such that only 3 or 6 are able to activate (the other class neurons are inhibited, indicated by the gray shading). Because the stimulus is inconsistent with 6, the network settled to 3 and reconstructed the left part of the digit.

The latter part of the experiment illustrates the integration of information between several partially specified cues, which is of interest for solving sensorimotor transformation or multi-modal sensory cue integration problems (Deneve et al., 2001; Doya

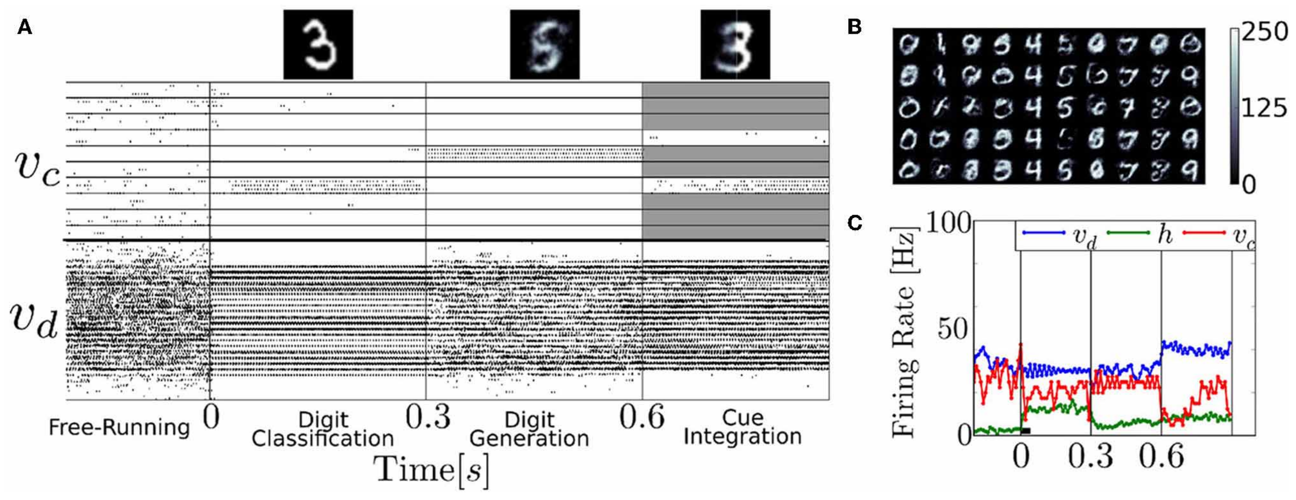


FIGURE 9 | The recurrent structure of the network allows it to classify, reconstruct and infer from partial evidence. (A) Raster plot of an experiment illustrating these features. Before time 0*s*, the neural RBM runs freely, with no input. Due to the stochasticity in the network, the activity wanders from attractor to attractor. At time 0*s*, the digit 3 is presented (i.e., layer v_d is driven by *d*), activating the correct class label in v_c; At time $t = 0.3$ s, the class neurons associated to 5 are clamped to high activity and the rest of the class label neurons are strongly inhibited, driving the network to reconstruct its version of the digit in layer v_d; At time $t = 0.6$ s, the right-half part of a digit 8 is presented, and the class neurons are stimulated such that only 3 or 6 can activate (all others are strongly inhibited as indicated by the gray shading). Because the stimulus is inconsistent with 6, the network settles to a 3 and attempts to reconstruct it. The top figures show the digits reconstructed in layer v_d. **(B)** Digits 0–9, reconstructed in the same manner. The columns correspond to clamping digits 0–9, and each is different, independent run. **(C)** Population firing rate of the experiment presented in **(A)**. The network activity is typically at equilibrium after about $10\tau_r = 40$ ms (black bar).

et al., 2006; Corneil et al., 2012). This feature has been used for auditory-visual sensory fusion in a spiking Deep Belief Network (DBN) model (O'Connor et al., 2013). There, the authors trained a DBN with visual and auditory data, which learned to associate the two sensory modalities, very similarly to how class labels and visual data are associated in our architecture. Their network was able to resolve a similar ambiguity as in our experiment in **Figure 9**, but using auditory inputs instead of a class label.

During digit generation, the trained network had a tendency to be globally bistable, whereby the layer v_d completely deactivated layer h. Since all the interactions between v_d and v_c take place through the hidden layer, v_c could not reconstruct the digit. To avoid this, we added populations of I&F neurons that were wired to layers v and h, respectively. The parameters of these neurons and their couplings were tuned such that each layer was strongly excited when it's average firing rate fell below 5 Hz.

4. DISCUSSION

Neuromorphic systems are promising alternatives for large-scale implementations of RBMs and deep networks, but the common procedure used to train such networks, Contrastive Divergence (CD), involves iterative, discrete-time updates that do not straightforwardly map on a neural substrate. We solve this problem in the context of the RBM with a spiking neural network model that uses the recurrent network dynamics to compute these updates in a continuous-time fashion. We argue that the recurrent activity coupled with STDP dynamics implements an event-driven variant of CD. Event-driven CD enables the system to learn on-line, while being able to carry out functionally relevant tasks such as recognition, data generation and cue integration.

The CD algorithm can be used to learn the parameters of probability distributions other than the Boltzmann distribution (even those without any symmetry assumptions). Our choice for the RBM, whose underlying probability distribution is a special case of the Boltzmann distribution, is motivated by the following facts: They are universal approximators of discrete distributions (Le Roux and Bengio, 2008); the conditions under which a spiking neural circuit can naturally perform MCMC sampling of a Boltzmann distribution were previously studied (Merolla et al., 2010; Buesing et al., 2011); and RBMs form the building blocks of many deep learning models such as DBNs, which achieve state-of-the-art performance in many machine learning tasks (Bengio, 2009). The ability to implement RBMs with spiking neurons and train then using event-based CD paves the way toward on-line training of DBNs of spiking neurons (Hinton et al., 2006).

We chose the MNIST handwritten digit task as a benchmark for testing our model. When the RBM was trained with standard CD, it could recognize up to 926 out of 1000 of out-of-training samples. The MNIST handwritten digit recognition task was previously shown in a digital neuromorphic chip (Arthur et al., 2012), which performed at 89% accuracy, and in a software simulated visual cortex model (Eliasmith et al., 2012). However, both implementations were configured using weights trained off-line. A recent article showed the mapping of off-line trained DBNs onto spiking neural network (O'Connor et al., 2013). Their results demonstrated hand-written digit recognition using neuromorphic event-based sensors as a source of input spikes. Their performance reached up to 94.1% using leaky I&F neurons. The use of an additional layer explains to a large extent their better performance compared to ours (91.9%). Our work extends (O'Connor et al., 2013) with on-line training that is based on synaptic plasticity, testing its robustness to finite weight precision, and providing an interpretation of spiking activity in terms of neural sampling.

To achieve the computations necessary for sampling from the RBM, we have used a neural sampling framework (Fiser et al., 2010), where each spike is interpreted as a sample of an underlying probability distribution. Buesing et al. proved that abstract neuron models consistent with the behavior of biological spiking neurons can perform MCMC, and have applied it to a basic learning task in a fully visible Boltzmann Machine. We extended the neural sampling framework in three ways: First, we identified the conditions under which a dynamical system consisting of I&F neurons can perform neural sampling; Second, we verified that the sampling of RBMs was robust to finite-precision parameters; Third, we demonstrated learning in a Boltzmann Machine with hidden units using STDP synapses.

In neural sampling, neurons behave stochastically. This behavior can be achieved in I&F neurons using noisy input currents, created by a Poisson spike train. Spike trains with Poisson-like statistics can be generated with no additional source of noise, for example by the following mechanisms: balanced excitatory and inhibitory connections (van Vreeswijk and Sompolinsky, 1996), finite-size effects in a large network, and neural mismatch (Amit and Brunel, 1997). The latter mechanism is particularly appealing, because it benefits from fabrication mismatch and operating noise inherent to neuromorphic implementations (Chicca and Fusi, 2001).

Other groups have also proposed to use I&F neuron models for computing the Boltzmann distribution. (Merolla et al., 2010) have shown that noisy I&F neurons' activation function is approximately a sigmoid as required by the Boltzmann machine, and have devised a scheme whereby a global inhibitory rhythm drives the network to generate samples of the Boltzmann distribution. O'Connor et al. (2013) have demonstrated a deep belief network of I&F neurons that was trained off-line, using standard CD and tested it using the MNIST database. Independently and simultaneously to this work, Petrovici et al. (2013) demonstrated that conductance-based I&F neurons in a noisy environment are compatible with neural sampling as described in Buesing et al. (2011). Similarly, Petrovici et al. find that the choice of non-rectangular PSPs and the approximations made by the I&F neurons are not critical to the performance of the neural sampler. Our work extends all of those above by providing an online, STDP-based learning rule to train RBMs sampled using I&F neurons.

4.1. APPLICABILITY TO NEUROMORPHIC HARDWARE

Neuromorphic systems are sensible to fabrication mismatch and operating noise. Fortunately, the mismatch in the synaptic weights and the activation function parameters γ and β are not an issue if the biases and the weights are learned, and the functionality of the RBM is robust to small variations in the weights caused by discretization. These two findings are encouraging for neuromorphic implementations of RBMs. However, at least two conceptual problems of the presented RBM architecture must be solved in order to implement such systems on a larger scale. First, the symmetry condition required by the RBM does not necessarily hold. In a neuromorphic device, the symmetry condition is impossible to guarantee if the synapse weights are stored locally at each neuron. Sharing one synapse circuit per pair of neurons can solve this problem. This may be impractical due to the very large number of synapse circuits in the network, but may be less problematic when using Resistive Random-Access Memorys (RRAMs) (also called *memristors*) crossbar arrays to emulate synapses (Kuzum et al., 2011; Cruz-Albrecht et al., 2013; Serrano-Gotarredona et al., 2013).RRAM are a new class of nanoscale devices whose current-voltage relationship depends on the history of other electrical quantities (Strukov et al., 2008), and so act like programmable resistors. Because they can conduct currents in both directions, one RRAM circuit can be shared between a pair of neurons. A second problem is the number of recurrent connections. Even our RBM of modest dimensions involved almost two million synapses, which is impractical in terms of bandwidth and weight storage. Even if a very high number of weights are zero, the connections between each pair of neurons must exist in order for a synapse to learn such weights. One possible solution is to impose sparse connectivity between the layers (Murray and Kreutz-Delgado, 2007; Tang and Eliasmith, 2010) and implement synaptic connectivity in a scalable hierarchical address-event routing architecture (Joshi et al., 2010; Park et al., 2012).

4.2. OUTLOOK: A CUSTOM LEARNING RULE

Our method combines I&F neurons that perform neural sampling and the CD rule. Although we showed that this leads to a functional model, we do not know whether event-driven CD is optimal in any sense. This is partly due to the fact that CD_k is an approximate rule (Hinton, 2002), and it is still not entirely understood why it performs so well, despite extensive work in studying its convergence properties (Carreira-Perpinan and Hinton, 2005). Furthermore, the distribution sampled by the I&F neuron does not exactly correspond to the Boltzmann distribution, and the average weight updates in event-driven CD differ from those of standard CD, because in the latter they are carried out at the end of the reconstruction step.

A very attractive alternative is to derive a custom synaptic plasticity rule that minimizes some functionally relevant quantity (such as Kullback-Leibler divergence or Contrastive Divergence), *given* the encoding of the information in the I&F neuron (Deneve, 2008; Brea et al., 2013). A similar idea was recently pursued in Brea et al. (2013), where the authors derived a triplet-based synaptic learning rule that minimizes an upper bound of the Kullback–Leibler divergence between the model and the data distributions. Interestingly, their rule had a similar global signal that modulates the learning rule, as in event-driven CD, although the nature of this resemblance remains to be explored. Such custom learning rules can be very beneficial in guiding the design of on-chip plasticity in neuromorphic VLSI and RRAM nanotechnologies, and will be the focus of future research.

ACKNOWLEDGMENTS

This work was partially funded by the National Science Foundation (NSF EFRI-1137279, CCF-1317560), the Office of

Naval Research (ONR MURI 14-13-1-0205), and the Swiss National Science Foundation (PA00P2_142058).

REFERENCES

Amit, D., and Brunel, N. (1997). Model of global spontaneous activity and local structured activity during delay periods in the cerebral cortex. *Cereb. Cortex* 7, 237–252. doi: 10.1093/cercor/7.3.237

Arthur, J., Merolla, P., Akopyan, F., Alvarez, R., Cassidy, A., Chandra, S., et al. (2012). "Building block of a programmable neuromorphic substrate: a digital neurosynaptic core," in *The 2012 International Joint Conference on Neural Networks (IJCNN)* (Brisbane, QLD: IEEE), 1–8. doi: 10.1109/IJCNN.2012.6252637

Bartolozzi, C., and Indiveri, G. (2007). Synaptic dynamics in analog VLSI. *Neural Comput.* 19, 2581–2603. doi: 10.1162/neco.2007.19.10.2581

Bengio, Y. (2009). Learning deep architectures for ai. *Found. Trends Mach. Learn.* 2, 1–127. doi: 10.1561/2200000006

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., et al. (2010). "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*. Vol. 4 (Austin, TX). Available online at: http://deeplearning.net/software/theano/

Brea, J., Senn, W., and Pfister, J.-P. (2013). Matching recall and storage in sequence learning with spiking neural networks. *J. Neurosci.* 33, 9565–9575. doi: 10.1523/JNEUROSCI.4098-12.2013

Buesing, L., Bill, J., Nessler, B., and Maass, W. (2011). Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput. Biol.* 7:e1002211. doi: 10.1371/journal.pcbi.1002211

Carreira-Perpinan, M. A., and Hinton, G. E. (2005). On contrastive divergence learning. *Artif. Intell. Stat.* 2005, 17. Available online at: http://www.gatsby.ucl.ac.uk/aistats/AIabst.htm

Chicca, E. and Fusi, S. (2001). "Stochastic synaptic plasticity in deterministic aVLSI networks of spiking neurons," in *Proceedings of the World Congress on Neuroinformatics, ARGESIM Reports*, ed. F. Rattay (Vienna: ARGESIM/ASIM Verlag), 468–477.

Corneil, D., Sonnleithner, D., Neftci, E., Chicca, E., Cook, M., Indiveri, G., et al. (2012). "Function approximation with uncertainty propagation in a VLSI spiking neural network," in *International Joint Conference on Neural Networks, IJCNN* (Brisbane: IEEE), 2990–2996. doi: 10.1109/IJCNN.2012.6252780

Cox, D. (1962). *Renewal Theory.* Vol. 1. London: Methuen.

Cruz-Albrecht, J. M., Derosier, T., and Srinivasa, N. (2013). A scalable neural chip with synaptic electronics using cmos integrated memristors. *Nanotechnology* 24, 384011. doi: 10.1088/0957-4484/24/38/384011

Deiss, S., Douglas, R., and Whatley, A. (1998). "A pulse-coded communications infrastructure for neuromorphic systems, chapter 6," in *Pulsed Neural Networks*, eds W. Maass and C. Bishop (Cambridge, MA: MIT Press), 157–178.

Deneve, S. (2008). Bayesian spiking neurons I: inference. *Neural Comput.* 20, 91–117. doi: 10.1162/neco.2008.20.1.91

Deneve, S., Latham, P., and Pouget, A. (2001). Efficient computation and cue integration with noisy population codes. *Nature Neurosci.* 4, 826–831. doi: 10.1038/90541

Destexhe, A., Mainen, Z., and Sejnowski, T. (1998). "Kinetic models of synaptic transmission ," in *Methods in Neuronal Modelling, from Ions to Networks*, eds C. Koch and I. Segev (Cambridge, MA: MIT Press), 1–25.

Doya, K., Ishii, S., Pouget, A., and Rao, R. (2006). *Bayesian Brain Probabilistic Approaches to Neural Coding*. Cambridge, MA: MIT Press. doi: 10.7551/mitpress/9780262042383.001.0001

Eliasmith, C., Stewart, T., Choo, X., Bekolay, T., DeWolf, T., Tang, Y., et al. (2012). A large-scale model of the functioning brain. *Science* 338, 1202–1205. doi: 10.1126/science.1225266

Fiser, J., Berkes, P., Orbán, G., and Lengyel, M. (2010). Statistically optimal perception and learning: from behavior to neural representations: perceptual learning, motor learning, and automaticity. *Trends Cogn. Sci.* 14, 119. doi: 10.1016/j.tics.2010.01.003

Fusi, S., and Mattia, M. (1999). Collective behavior of networks with linear (VLSI) integrate and fire neurons. *Neural Comput.* 11, 633–652. doi: 10.1162/089976699300016601

Gardiner, C. W. (2012). *Handbook of Stochastic Methods*. Berlin: Springer. doi: 10.1007/978-3-662-02377-8

Gerstner, W., and Kistler, W. (2002). *Spiking Neuron Models. Single Neurons, Populations, Plasticity*. Cambridge: Cambridge University Press. doi: 10.1017/CBO9780511815706

Goodman, D., and Brette, R. (2008). Brian: a simulator for spiking neural networks in Python. *Front. Neuroinform.* 2:5. doi: 10.3389/neuro.11.005.2008

Griffiths, T., Chater, N., Kemp, C., Perfors, A., and Tenenbaum, J. B. (2010). Probabilistic models of cognition: exploring representations and inductive biases. *Trends Cogn. Sci.* 14, 357–364. doi: 10.1016/j.tics.2010.05.004

Haykin, S. (1998). *Neural Networks: A Comprehensive Foundation*. 2nd Edn. Prentice Hall. Available online at: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0132733501

Hinton, G., Osindero, S., and Teh, Y. (2006). A fast learning algorithm for deep belief nets. *Neural Comput.* 18, 1527–1554. doi: 10.1162/neco.2006.18.7.1527

Hinton, G., and Salakhutdinov, R. (2006). Reducing the dimensionality of data with neural networks. *Science* 313, 504–507. doi: 10.1126/science.1127647

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Comput.* 14, 1771–1800. doi: 10.1162/089976602760128018

Indiveri, G., Linares-Barranco, B., Hamilton, T., van Schaik, A., Etienne-Cummings, R., Delbruck, T., et al. (2011). Neuromorphic silicon neuron circuits. *Front. Neurosci.* 5, 1–23. doi: 10.3389/fnins.2011.00073

Joshi, S., Deiss, S., Arnold, M., Park, J., Yu, T., and Cauwenberghs, G. (2010). "Scalable event routing in hierarchical neural array architecture with global synaptic connectivity," in *12th International Workshop on Cellular Nanoscale Networks and Their Applications* (Berkeley, CA: IEEE), 1–6. doi: 10.1109/CNNA.2010.5430296

Kempter, R., Gerstner, W., and Van Hemmen, J. (2001). Intrinsic stabilization of output rates by spike-based hebbian learning. *Neural Comput.* 13, 2709–2741. doi: 10.1162/089976601317098501

Kuzum, D., Jeyasingh, R. G., Lee, B., and Wong, H.-S. P. (2011). Nanoelectronic programmable synapses based on phase change materials for brain-inspired computing. *Nano Lett.* 12, 2179–2186. doi: 10.1021/nl201040y

Le, Q. V., Ranzato, M., Monga, R., Devin, M., Chen, K., Corrado, G. S., et al. (2011). Building high-level features using large scale unsupervised learning. arXiv preprint: arXiv:1112.6209.

Le Roux, N., and Bengio, Y. (2008). Representational power of restricted boltzmann machines and deep belief networks. *Neural Comput.* 20, 1631–1649. doi: 10.1162/neco.2008.04-07-510

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proc. IEEE* 86, 2278–2324. doi: 10.1109/5.726791

Liu, S.-C., and Delbruck, T. (2010). Neuromorphic sensory systems. *Curr. Opin. Neurobiol.* 20, 288–295. doi: 10.1016/j.conb.2010.03.007

Mead, C. (1989). *Analog VLSI and Neural Systems*. Reading, MA: Addison-Wesley. doi: 10.1007/978-1-4613-1639-8

Merolla, P., Ursell, T., and Arthur, J. (2010). The thermodynamic temperature of a rhythmic spiking network. *CoRR*. abs/1009.5473, ArXiv e-prints. Available online at: http://arxiv.org/abs/1009.5473

Murray, J. F., and Kreutz-Delgado, K. (2007). Visual recognition and inference using dynamic over complete sparse learning. *Neural Comput.* 19, 2301–2352. doi: 10.1162/neco.2007.19.9.2301

Neftci, E., Binas, J., Rutishauser, U., Chicca, E., Indiveri, G., and Douglas, R. J. (2013). Synthesizing cognition in neuromorphic electronic systems. *Proc. Natl. Acad. Sci. U.S.A.* 110, E3468–E3476. doi: 10.1073/pnas.1212083110

Neftci, E., Toth, B., Indiveri, G., and Abarbanel, H. (2012). Dynamic state and parameter estimation applied to neuromorphic systems. *Neural Comput.* 24, 1669–1694. doi: 10.1162/NECO_a_00293

O'Connor, P., Neil, D., Liu, S.-C., Delbruck, T., and Pfeiffer, M. (2013). Real-time classification and sensor fusion with a spiking deep belief network. *Front. Neurosci.* 7:178. doi: 10.3389/fnins.2013.00178

Park, J., Yu, T., Maier, C., Joshi, S., and Cauwenberghs, G. (2012). "Live demonstration: Hierarchical address-event routing architecture for reconfigurable large scale neuromorphic systems," in *Circuits and Systems (ISCAS), 2012 IEEE International Symposium on*, (Seoul), 707, 711, 20–23. doi: 10.1109/ISCAS.2012.6272133. Available online at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6272133

Pedroni, B., Das, S., Neftci, E., Kreutz-Delgado, K., and Cauwenberghs, G. (2013). "Neuromorphic adaptations of restricted boltzmann machines and deep belief networks," in *International Joint Conference on Neural Networks, IJCNN*. (Dallas, TX).

Petrovici, M. A., Bill, J., Bytschok, I., Schemmel, J., and Meier, K. (2013). Stochastic inference with deterministic spiking neurons. arXiv preprint: arXiv:1311. 3211.

Plesser, H. E., and Gerstner, W. (2000). Noise in integrate-and-fire neurons: from stochastic input to escape rates. *Neural Comput.* 12, 367–384. doi: 10.1162/089976600300015835

Renart, A., Song, P., and Wang, X.-J. (2003). Robust spatial working memory through homeostatic synaptic scaling in heterogeneous cortical networks. *Neuron* 38, 473–485. doi: 10.1016/S0896-6273(03)00255-1

Robbins, H., and Monro, S. (1951). A stochastic approximation method. *Ann. Math. Stat.* 22, 400–407. doi: 10.1214/aoms/1177729586

Schemmel, J., Brüderle, D., Grübl, A., Hock, M., Meier, K., and Millner, S. (2010). "A wafer-scale neuromorphic hardware system for large-scale neural modeling," in *International Symposium on Circuits and Systems, ISCAS* (Paris: IEEE), 1947–1950. doi: 10.1109/ISCAS.2010.5536970

Serrano-Gotarredona, T., Masquelier, T., Prodromakis, T., Indiveri, G., and Linares-Barranco, B. (2013). Stdp and stdp variations with memristors for spiking neuromorphic learning systems. *Front. Neurosci.* 7:2. doi: 10.3389/fnins.2013.00002

Silver, R., Boahen, K., Grillner, S., Kopell, N., and Olsen, K. (2007). Neurotech for neuroscience: unifying concepts, organizing principles, and emerging tools. *J. Neurosci.* 27, 11807. doi: 10.1523/JNEUROSCI.3575-07.2007

Strukov, D. B., Snider, G. S., Stewart, D. R., and Williams, R. S. (2008). The missing memristor found. *Nature* 453, 80–83. doi: 10.1038/nature06932

Tang, Y. and Eliasmith, C. (2010). "Deep networks for robust visual recognition," in *Proceedings of the 27th International Conference on Machine Learning (ICML-10)* (Haifa), 1055–1062. Available online at: http://www.icml2010.org/papers/370.pdf

van Vreeswijk, C., and Sompolinsky, H. (1996). Chaos in neuronal networks with balanced excitatory and inhibitory activity. *Science* 274, 1724–1726. doi: 10.1126/science.274.5293.1724

Yu, T., Park, J., Joshi, S., Maier, C., and Cauwenberghs, G. (2012) "65k-neuron integrate-and-fire array transceiver with address-event reconfigurable synaptic routing," in *Biomedical Circuits and Systems Conference (BioCAS), IEEE*, (Hsinch), 21, 24. 28–30. doi: 10.1109/BioCAS.2012.6418479. Available online at: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6418479

Conflict of Interest Statement: The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Received: 07 October 2013; accepted: 22 December 2013; published online: 30 January 2014.

Citation: Neftci E, Das S, Pedroni B, Kreutz-Delgado K and Cauwenberghs G (2014) Event-driven contrastive divergence for spiking neuromorphic systems. Front. Neurosci. ***7****:272. doi: 10.3389/fnins.2013.00272*

This article was submitted to Neuromorphic Engineering, a section of the journal Frontiers in Neuroscience.