Spiking Neural Networks –
Part II: Detecting Spatio-Temporal Patterns

Nicolas Skatchkovsky*, Hyeryung Jang* and Osvaldo Simeone

Abstract—Inspired by the operation of biological brains, Spiking Neural Networks (SNNs) have the unique ability to detect information encoded in spatio-temporal patterns of spiking signals. Examples of data types requiring spatio-temporal processing include logs of time stamps, e.g., of tweets, and outputs of neural prostheses and neuromorphic sensors. In this paper, the second of a series of three review papers on SNNs, we first review models and training algorithms for the dominant approach that considers SNNs as a Recurrent Neural Network (RNN) and adapt learning rules based on backpropagation through time to the requirements of SNNs. In order to tackle the non-differentiability of the spiking mechanism, state-of-the-art solutions use surrogate gradients that approximate the threshold activation function with a differentiable function. Then, we describe an alternative approach that relies on probabilistic models for spiking neurons, allowing the derivation of local learning rules via stochastic estimates of the gradient. Finally, experiments are provided for neuromorphic data sets, yielding insights on accuracy and convergence under different SNN models.

I. INTRODUCTION

Spiking Neural Networks (SNNs) have recently emerged as a more energy efficient alternative to Artificial Neural Networks (ANNs) for applications involving on-device learning and inference. SNNs are inspired by the operation of biological brains, and function using sparse, binary *spiking* signals. Unlike the static, real-valued, signals used to train ANNs, spiking signals encode information using the *spatio-temporal* patterns defined by spike timings. Thanks to efficient hardware implementations, SNNs often may unlock a new class of applications for mobile or embedded devices.

Training ANNs relies on the backpropagation (BP) algorithm, which carries out "credit assignment" for the output loss to the network by implementing the chain rule of differentiation. Unlike for ANNs, the gradient of the output loss does not provide an informative learning signal for SNNs. In fact, spiking neurons are characterized by threshold activation function, whose derivative is zero almost everywhere. Different lines of research have been pursued to overcome this challenge. As we reviewed in Part I, a first approach bypasses the problem by training an ANN for a specific task, and then converting its weights to a SNN at inference time. When employing such techniques, SNNs are used to detect spatial patterns from data,

hence implementing the functionality of ANNs [1]–[3], albeit with energy consumption improvements.

A major argument in favor of the use of SNNs is their ability to process information encoded both over space and over time (see Fig. 2). The slow progress in the development of training algorithms for SNNs that are able to detect spatio-temporal patterns stems in part lack of widely accepted benchmark *neuromorphic* data sets [4]. The recent emergence of data sets captured using neuromorphic sensors [4]–[7] is quickly changing the research landscape, leading to a number of breakthroughs, while still leaving important open questions.

Training SNNs for detecting spatio-temporal patterns has followed into two main approaches. The first, currently dominant, class of methods views SNNs as a subclass of Recurrent Neural Networks (RNNs), and adapts standard algorithms for RNNs such as backpropagation through time (BPTT) and forward-mode differentiation to the unique features of SNNs. In order to tackle the issues due to non differentiable activation function, state-of-the-art solutions are given by surrogate gradient (SG) algorithms [8] that smooth out the derivative with a differentiable surrogate function, and/or by approximating the learning signals with either random feedback signals [8], [9] or with locally computed surrogate losses [10].

The second line of work characterizes the operation of SNNs in a probabilistic manner by introducing randomness in the spiking mechanism [11]–[13]. A probabilistic model defines differentiable output loss and enables the direct derivation of learning algorithms using principled learning criteria, which do not require the implementation of backpropagation mechanisms.

The remainder of this paper is organized as follows. In Sec. II, we provide a review of SNN models used for detecting spatio-temporal patterns, then introduce the corresponding learning algorithms in Sec. III. In Sec. IV, we explore an application for classification task. In conclusion, we introduce extensions in Sec. V.

II. DETECTING SPATIO-TEMPORAL PATTERNS: MODELS

In this section, we first review the details of the deterministic discrete-time Spike Response Model (SRM) introduced in Part I, and then describe a probabilistic extension of the SRM known as Generalized Linear Model (GLM). Throughout this paper, we consider a general SNN architecture described by a directed, possibly cyclic, graph with spiking neurons as vertices and synapses as directed edges. As seen in Fig. 1, each neuron i receives synaptic connections from a subset $\mathcal{P}_i$ of other spiking neurons. With a slight abuse of notations, we take the set $\mathcal{P}_i$ to include also exogenous inputs.
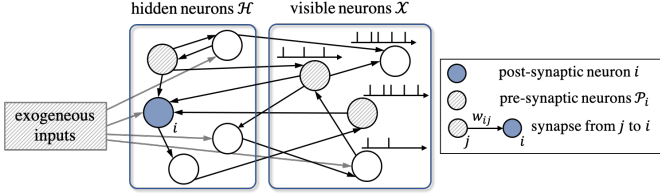
Fig. 1. Architecture of an SNN with exogenous inputs, $|\mathcal{X}| = 4$ visible neurons, and $|\mathcal{H}| = 5$ hidden neurons. Synapses are represented by directed links between two neurons.

A. Deterministic Spike Response Model (SRM)

As described in Part I, under the discrete-time SRM, the binary output $s_{i,t} \in \{0, 1\}$ of spiking neuron $i \in \mathcal{V}$ at time t depends on its membrane potential $u_{i,t}$ as

$$s_{i,t} = \Theta(u_{i,t} - \vartheta), \qquad (1)$$

where $\Theta(\cdot)$ is the Heaviside step function. Accordingly, a spike is emitted when the membrane potential $u_{i,t}$ crosses a fixed threshold ϑ, where the membrane potential is obtained as

$$u_{i,t} = \sum_{j \in \mathcal{P}_i} w_{ij} \big(\underbrace{\alpha_t * s_{j,t}}_{:= p_{j,t}} \big) + \big(\underbrace{\beta_t * s_{i,t}}_{:= r_{i,t}} \big) + \gamma_i, \qquad (2)$$

where w_{ij} is the synaptic weight from pre-synaptic neuron $j \in \mathcal{P}_i$ to post-synaptic neuron i; α_t and β_t represent the synaptic and feedback spike responses, respectively; and $*$ denotes the convolution operator. We collect in vector $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i \in \mathcal{V}}$ the model parameters, with $\boldsymbol{\theta}_i := \{w_{ij}\}_{j \in \mathcal{P}_i}$ being the local model parameters of each neuron i. The bias is typically set to $\gamma_i = 0$.

SRM is commonly implemented with the *alpha-function* spike response $\alpha_t = \exp(-t/\tau_{\mathrm{mem}}) - \exp(-t/\tau_{\mathrm{syn}})$ and the exponentially decaying feedback filter $\beta_t = -\exp(-t/\tau_{\mathrm{ref}})$ for $t \geq 1$ with some positive constants $\tau_{\mathrm{mem}}, \tau_{\mathrm{syn}}$, and τ_{ref}. With these choices, the filtered contributions in (2) can be computed using the recursive equations

$$p_{j,t} = \exp\big(-1/\tau_{\mathrm{mem}}\big)p_{j,t-1} + q_{j,t-1}, \qquad (3a)$$

$$\text{with} \quad q_{j,t} = \exp\big(-1/\tau_{\mathrm{syn}}\big)q_{j,t-1} + s_{j,t-1}, \qquad (3b)$$

$$\text{and} \quad r_{i,t} = \exp\big(-1/\tau_{\mathrm{ref}}\big)r_{i,t-1} + s_{i,t-1}. \qquad (3c)$$

Using (3a)-(3b), each synapse (j, i) between pre-synaptic neuron j and post-synaptic neuron i computes the synaptic filtered trace $p_{j,t}$ in (2) via a second-order autoregressive (AR) filter, while the soma of each neuron i computes feedback filtered trace $r_{i,t}$ in (2) via a first-order AR filter. The dynamics of spiking neurons (1)-(3) define the operation of a Recurrent Neural Network (RNN) with threshold activation function (1) and binary activations [8], [14].

B. Probabilistic Generalized Linear Model (GLM)

A probabilistic generalization of the SRM used in computational neuroscience is the Generalized Linear Model (GLM), which introduces randomness in the spiking mechanism (1). Define as $\boldsymbol{s}_t = (s_{i,t} : i \in \mathcal{V})$ the spiking signals for all neurons at time t and as $\boldsymbol{s}_{\leq t-1} = (\boldsymbol{s}_1, \ldots, \boldsymbol{s}_{t-1})$ all signals up to time $t - 1$. Under GLMs, the spiking probability of neuron i conditioned on $\boldsymbol{s}_{\leq t-1}$ is given as [11]–[13]

$$p_{\boldsymbol{\theta}_i}(s_{i,t} = 1 | \boldsymbol{s}_{\leq t-1}) = p_{\boldsymbol{\theta}_i}(s_{i,t} = 1 | u_{i,t}) = \sigma_\Delta(u_{i,t}), \quad (4)$$

where $\sigma_\Delta(x) = 1/(1 + \exp(-\Delta x))$ is the sigmoid function, $\Delta > 0$ is a bandwidth parameter, and the membrane potential $u_{i,t}$ is defined in (2). Conditioned on $\boldsymbol{s}_{\leq t-1}$, the variables $s_{i,t}$ are independent across the neuron index i. As per (4), a larger membrane potential increases the spiking probability. The notation used in (4) emphasizes the dependence on the local model parameters $\boldsymbol{\theta}_i = \{\{w_{ij}\}_{j \in \mathcal{P}_i}, \gamma_i\}$ of neuron i. The GLM (4) reduces to the SRM model (1) when $\Delta \to \infty$ and $\gamma_i = -\vartheta$ so that the sigmoid function $\sigma_\Delta(u_{i,t})$ tends to the hard threshold function $\Theta(u_{i,t} - \vartheta)$. Throughout the paper, we simply denote the sigmoid function by $\sigma(\cdot)$.

By (4), the negative log-probability of the spike is given by

$$-\log p_{\boldsymbol{\theta}_i}(s_{i,t} | u_{i,t}) = \ell\big(s_{i,t}, \sigma(u_{i,t})\big), \qquad (5)$$

where $\ell(a, b) = -a \log(b) - (1 - a) \log(1 - b)$ is the binary cross-entropy loss function. Using the chain rule, the negative log-probability of the spike sequence $\boldsymbol{s}_{\leq T}$ emitted by all neurons up to time T can be written as $-\log p_{\boldsymbol{\theta}}(\boldsymbol{s}_{\leq T}) = \sum_{t=1}^{T} \sum_{i \in \mathcal{V}} \ell\big(s_{i,t}, \sigma(u_{i,t})\big)$. As a remark, in GLMs, it is common to associate each synapse with multiple spike responses and corresponding weights, allowing to learn distinct temporal receptive fields across the spiking neurons [11], [13].

For any two jointly distributed random process $\boldsymbol{a}_{\leq T}$ and $\boldsymbol{b}_{\leq T}$, we introduce the causal conditioning notation [16] of $\boldsymbol{a}_{\leq T}$ given $\boldsymbol{b}_{\leq T}$ denoted by $p(\mathbf{a}_{\leq T} || \mathbf{b}_{\leq T}) = \prod_{t=1}^{T} p(\mathbf{a}_t | \mathbf{a}_{\leq t-1}, \mathbf{b}_{\leq t})$.

III. DETECTING SPATIO-TEMPORAL PATTERNS: ALGORITHMS

In this section, we consider the problem of training SNNs using either supervised learning or unsupervised generative learning for the purpose of detecting, or generating, spatio-temporal patterns of spiking signals. Theses are encoded in target spiking outputs for a subset $\mathcal{X} \subseteq \mathcal{V}$ of neurons, referred to as "visible" (see Fig. 1). In supervised learning, visible neurons represent the read-out layer that encodes the index of the desired class for classification or a continuous target variable for regression. Information encoding into spiking signals may be based on rate, time, and population encoding principles (see Fig. 2). In unsupervised generative learning, the output of the visible neurons directly provides the desired signals to be generated by the SNN.

Training relies on the availability of a training data set $\mathcal{D}$ containing $|\mathcal{D}|$ examples of spiking signals $\boldsymbol{x}_{\leq T}$ of duration T samples for the subset $\mathcal{X}$ of visible neurons. The spiking signals $\boldsymbol{h}_{\leq T}$ of the complementary subset $\mathcal{H} = \mathcal{V} \setminus \mathcal{X}$ of *hidden*, or latent, neurons are not specified by the training set, and their behavior should be adapted during training to ensure the desired behavior of visible neurons. The error between the target spiking signals $\boldsymbol{x}_{\leq T} \in \mathcal{D}$ and the actual, possibly probabilistic, output spiking signals produced by the SNN is measured by a loss function that is defined differently for SRMs and GLMs. Defining as $\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$ the loss measured on the example $\boldsymbol{x}_{\leq T} \in \mathcal{D}$ for an SNN with model parameters $\boldsymbol{\theta}$, standard stochastic gradient-based learning rules update the model parameters $\boldsymbol{\theta}$ in the opposite direction of the gradient $\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$ as $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \nabla_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$, where η is the learning rate.
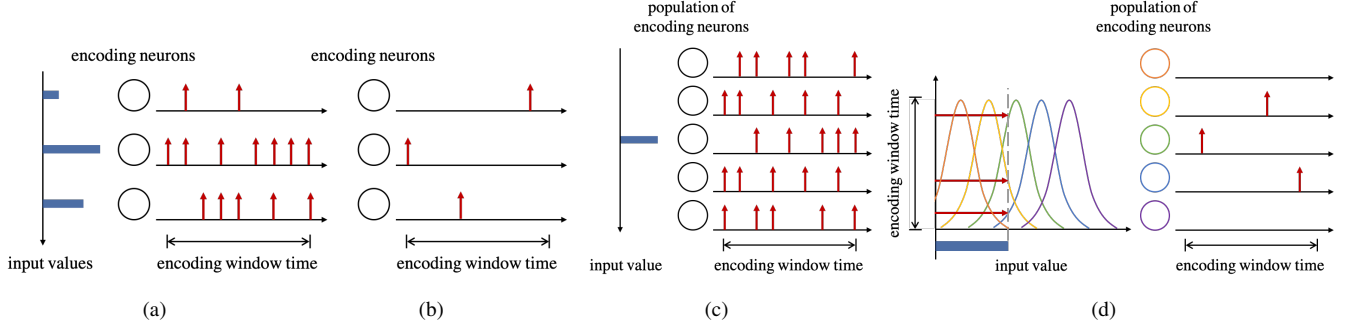
Fig. 2. Illustration of rate, time, and population encoding principles: (a) With rate coding, each input value, as shown using the horizontal blue bars on the left, is encoded in the spike rate of the corresponding encoding neuron: A larger input generates a large number of spikes within a fixed encoding window time; (b) With time encoding, an input value is encoded by the timing of one or more spikes emitted by the corresponding encoding neuron; (c) With population rate encoding, an input value is encoded by the average spike rate for a set of encoding neurons; (d) With population time encoding, an input value is encoded by at most one spike for each of the encoding neurons, whose timings are given by a "receptive field" function shown on the left [15].

A. Training SRM-Based Models

As compared to conventional methods for ANNs, applying gradient-based rules to training SRM-based SNNs poses a number of new challenges. First, the derivative of the activation function (1) is zero almost everywhere, making the gradient not informative for training. Second, as seen, SNNs are recurrent architectures due to refractoriness mechanism and, possibly, to cyclic directed topologies. SNNs hence inherit from Recurrent Neural Networks (RNNs) the problem of carrying out spatial (across neurons) and temporal (across time) *credit assignment* for the training loss. Finally, as required in most neuromorphic hardware platforms, training algorithms should only involve *local operations* along with minimal global signaling, making the standard backpropagation through time (BPTT) RNN training algorithm not desirable for SNNs.

A state-of-the-art approach to tackle the first problem is given by surrogate gradient (SG) methods. SG algorithms smooth out the Heaviside function in (1) when computing the gradient $\nabla_{\boldsymbol{\theta}} \mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$. To elaborate, assume that the loss function is written as a sum over $t = 1, \ldots, T$ and $i \in \mathcal{X}$ as

$$\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta}) = \sum_{t=1}^{T} \sum_{i \in \mathcal{X}} L(x_{i,t}, \bar{x}_{i,t}), \tag{6}$$

where each term $L(x_{i,t}, \bar{x}_{i,t})$ is a local loss measure for visible neuron $i \in \mathcal{X}$ at time t that depends only on the target output $x_{i,t}$ of neuron i at time t and on the actual outputs $\bar{x}_{i,t}$ of the same neuron. Note that the output $\bar{x}_{i,t}$ depends on the model parameters $\boldsymbol{\theta}$ (from (1)-(3)). Equation (6) holds for most standard error measures such as the *van Rossum distance* between $\boldsymbol{x}_{\leq T}$ and $\bar{\boldsymbol{x}}_{\leq T}$, which is defined as $L(x_t, \bar{x}_t) = \sum_{t=1}^{T} ||\epsilon_t * (x_t - \bar{x}_t)||^2$, with $\epsilon_t = \exp(-t/\tau_{\text{VR}})$ being an exponential decaying function parametrized by a time constant τ_{VR}. With this decomposition (6), the partial derivative of the training loss $\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$ with respect to the synaptic weight w_{ij} can be computed as

$$\frac{\partial \mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})}{\partial w_{ij}} = \sum_{t=1}^{T} \underbrace{\frac{\partial L(x_{i,t}, \bar{x}_{i,t})}{\partial \bar{x}_{i,t}}}_{e_{i,t}} \cdot \underbrace{\frac{\partial \bar{x}_{i,t}}{\partial u_{i,t}}}_{\Theta'(u_{i,t} - \vartheta)} \cdot \underbrace{\frac{\partial u_{i,t}}{\partial w_{ij}}}_{\alpha_t * s_{j,t}}, \tag{7}$$

where the first term $e_{i,t} := \partial L(x_{i,t}, \bar{x}_{i,t})/\partial \bar{x}_{i,t}$ is the deriva-

tive of the loss at time t with respect to the post-synaptic neuron i's output at time t; the second term is problematic since it includes the derivative of the threshold function $\Theta'(\cdot)$ from (1), which is zero almost everywhere; and the third term can be directly computed from (3) as the filtered pre-synaptic trace $\partial u_{i,t}/\partial w_{ij} = p_{j,t} = \alpha_t * s_{j,t}$. Following the SG approach, the derivative $\Theta'(\cdot)$ is substituted with the derivative of a differentiable surrogate function, such as rectifier [17], sigmoid [9], [18], or exponential function [19]. For example, with a sigmoid surrogate, we have $\partial \bar{x}_{i,t}/\partial u_{i,t} \approx \sigma'(u_{i,t} - \vartheta)$.

From (7), the resulting SG-based rule is given as

$$w_{ij} \leftarrow w_{ij} - \eta \sum_{t=1}^{T} \Delta w_{ij,t}. \tag{8}$$

with

$$\Delta w_{ij,t} = \underbrace{e_{i,t}}_{\text{error signal}} \underbrace{\sigma'(u_{i,t} - \vartheta)}_{\text{post}_{i,t}} \underbrace{(\alpha_t * s_{j,t})}_{\text{pre}_{j,t}}. \tag{9}$$

The update (8) implements a *three-factor* rule [20] in that the contribution $\Delta w_{ij,t}$ of each time t to the update of a synaptic weight w_{ij} is given by the product of a local pre-synaptic term $\text{pre}_{j,t}$, which is a function of the activity of the pre-synaptic neuron j; a local post-synaptic term $\text{post}_{i,t}$, which measures the current "sensitivity" to changes in the membrane potential of post-synaptic neuron i; and a *per-neuron* error signal $e_{i,t}$ that depends on the overall operation of the system. The computation of this term generally requires BPTT, or, alternatively, forward-mode differentiation [8]. Effective state-of-the-art solutions tackle the second and third challenges mentioned earlier by approximating the error $e_{i,t}$ with either random projections of the error signals measured at the visible neurons [8], [9] or with locally computed error signals [10].

B. Training GLM-Based SNNs

The most common learning criterion for probabilistic models is Maximum Likelihood (ML) [21]. ML selects model parameters $\boldsymbol{\theta}$ that maximize the log-probability that the set $\mathcal{X}$ of visible neurons outputs desired spiking signals $\boldsymbol{x}_{\leq T} \in \mathcal{D}$. Equivalently, the training loss $\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$ is given by the negative log-likelihood (NLL) $\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta}) = -\log p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\leq T}) = -\log \sum_{\boldsymbol{h}_{\leq T}} p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\leq T}, \boldsymbol{h}_{\leq T})$ that requires to average out the

stochastic signals $\boldsymbol{h}_{\leq T}$ produced by the hidden neurons. Using the causal conditioning notation (see Sec. II-B), the loss is given as

$$\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta}) = -\log \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{h}_{\leq T}||\boldsymbol{x}_{\leq T})}\big[p_{\boldsymbol{\theta}}(\boldsymbol{x}_{\leq T}||\boldsymbol{h}_{\leq T-1})\big]. \quad (10)$$

The expectation in (10) causes two problems. First, this expectation is generally intractable since its complexity is exponential in the product $|\mathcal{H}|T$. Second, the loss $\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$ cannot be factorized over neurons $i \in \mathcal{X}$ and time t as in (6).

To cope with outlined problems, a common approach is to upper bound the NLL using Jensen's inequality as

$$\mathcal{L}_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta}) \leq \mathbb{E}_{p_{\boldsymbol{\theta}}(\boldsymbol{h}_{\leq T}||\boldsymbol{x}_{\leq T})}\left[\sum_{t=1}^{T}\sum_{i\in\mathcal{X}}\ell\big(x_{i,t},\sigma(u_{i,t})\big)\right]$$
$$:= L_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta}), \quad (11)$$

where the contribution to the loss for each visible neuron $i \in \mathcal{X}$ at time t is measured locally by the cross entropy $\ell\big(x_{i,t},\sigma(u_{i,t})\big)$. Note that the membrane potential $u_{i,t}$ generally depends on the output of the hidden neurons. An empirical estimate of the gradient $\nabla_{\boldsymbol{\theta}} L_{\boldsymbol{x}_{\leq T}}(\boldsymbol{\theta})$ can be now obtained by replacing the expectation in (11) with a single sample of the hidden neurons' signals from the distribution $p_{\boldsymbol{\theta}}(\boldsymbol{h}_{\leq T}||\boldsymbol{x}_{\leq T})$.

For each visible neuron $i \in \mathcal{X}$, the resulting learning rule for synaptic weight w_{ij} can be directly computed as (8), with

$$\Delta w_{ij,t} = \big(\underbrace{x_{i,t} - \sigma(u_{i,t})}_{\text{post}_{i,t}}\big)\big(\underbrace{\alpha_t * s_{j,t}}_{\text{pre}_{j,t}}\big). \quad (12a)$$

For each hidden neuron $i \in \mathcal{H}$, one needs to deal with the additional complication that the distribution of the hidden neurons' outputs $p_{\boldsymbol{\theta}}(\boldsymbol{h}_{\leq T}||\boldsymbol{x}_{\leq T})$ depend on the parameter of the hidden neurons, preventing a direct empirical approximation as in (12a). To address this issue, we can resort to one of several gradient estimates [22], [23]. A notable example is the REINFORCE gradient, which yields

$$\Delta w_{ij,t} = \underbrace{\bar{e}_t}_{\text{global error}} \big(\underbrace{h_{i,t} - \sigma(u_{i,t})}_{\text{post}_{i,t}}\big)\big(\underbrace{\alpha_t * s_{j,t}}_{\text{pre}_{j,t}}\big), \quad (12b)$$

where the update (12a) for visible neurons yields a two-factor rule in that the contribution $\Delta w_{ij,t}$ of each time t is given by the product of a pre-synaptic trace $p_{j,t} = \alpha_t * s_{j,t}$, and a post-synaptic term $x_{i,t} - \sigma(u_{i,t})$. Unlike the sensitivity term in (9), the post-synaptic term in (12a) provides an error measure between the desired target and the average model output. In contrast, update $\Delta w_{ij,t}$ in (12b) yields a three-factor rule, as it involves also a *common* error signal $\bar{e}_t := \sum_{i\in\mathcal{X}}\ell\big(x_{i,t},\sigma(u_{i,t})\big)$. The post-synaptic term $\text{post}_{i,t}$ in (12b) for the hidden neurons can be interpreted as a predictive error between the best (in the least square sense) prediction of the model, namely $\sigma(u_{i,t})$, and the actual realized output $h_{i,t}$ of neuron $i \in \mathcal{H}$ at time t. Furthermore, the update (12b) for hidden neurons contains the error signal $\bar{e}_t$ that is necessary to guide the update of hidden neurons given the lack of a specific target for their outputs in the data.

Comparing the updates (9) and (12) for SRM and GLM, respectively, we highlight several differences. First, the operation

TABLE I
TEST ACCURACY OF PROBABILISTIC SNNS AND DECOLLE ON MNIST-DVS. PERIOD REFERS TO THE INVERSE OF THE SAMPLING RATE. THE ASTERISK $*$ INDICATES A CONVOLUTIONAL ARCHITECTURE.

MODEL	PERIOD	N_H	INPUT	ACC.
DECOLLE	1 MS	39,232*	PER-SIGN	99.4%
	1 MS	19,616*	BINARY	98.9%
	1 MS	512	PER-SIGN	86.8%
	10 MS	256	PER-SIGN	73.8%
	25 MS	256	PER-SIGN	65.8%
SNN	25 MS	512	PER-SIGN	83.50%
	25 MS	512	BINARY	80.80%
	25 MS	256	PER-SIGN	82.80%
	25 MS	256	BINARY	79.3%

of the system is probabilistic for GLM and deterministic for SRM. Second, as mentioned, the post-synaptic terms in (12) represent error signals rather than sensitivity measures. Third, only the hidden neurons implement three-factor rules. Fourth, the error signal in (9) is individual, i.e., per neuron, while it is common to all hidden neurons in (12b). The presence of a per-neuron error $e_{i,t}$ in (9), instead of a global error $\bar{e}_t$ in (12), may result in a faster convergence for (9) if the error signal $e_{i,t}$ is more informative about the derivative of the loss [8]. However, an exact evaluation of the error $e_{i,t}$ requires complex mechanisms, such as BPTT, and the effectiveness of the discussed approximations should be evaluated on a case-by-case basis.

IV. EXAMPLES

In this section, we compare SRM and GLM-based models in terms of their capacity to classify spatio-temporal patterns. To this end, we focus on the neuromorphic datasets MNIST-DVS [24] and DVSGesture [6]. The former consists of the recordings via a neuromorphic camera of $10,000$ images from the MNIST dataset displayed on a moving screen. Changes in the intensity of the scene result in pixels taking values in the set $\{-1,\ 0,\ +1\}$, where "-1" indicates a decrease in luminosity, "$+1$" an increase, and "0" no change. We crop the images to 26×26 pixels in the center of the images. This produces 676 input spiking signals when we discard information about the polarity of the spikes; or $1,352$ signals when we include two separate inputs per pixel for positive and negative spikes. The duration of the examples is limited to 2 seconds, and various subsampling periods are considered. The DVSGesture dataset consists of 11 hand gestures displayed to a neuromorphic camera in different lighting conditions. In this case, images are reduced to 32×32 pixels, yielding $1,024$ input spiking signals when discarding polarity, or $2,048$ with per-sign inputs.

To train SRM-based models, we adopt the surrogate gradient scheme DECOLLE [10], which uses per-layer random error signals as discussed in Sec. III-A; while GLM-based SNNs are trained as explained in Sec. II-B. SNNs are equipped with $N = N_H + N_V$ neurons, where the number N_V of output neurons is equal to the number of classes. For GLM-based SNNs, we assume a fully connected topology; while for SRM-based SNNs, we follow [10] and consider a layered architecture. In particular, for SRM, we adopt either a two-layer fully
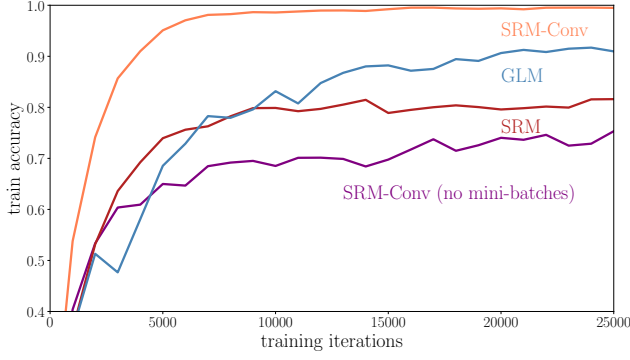
Fig. 3. Convergence speed of probabilistic GLM and SRM models, with the latter trained via trained with DECOLLE. For SRM-based SNNs, we include results with the convolutional architecture used in [10].

connected architecture with each layer having $N_H/2$ neurons, or, for reference, the convolutional architecture detailed in [10]. Note that the latter generally requires non-local updates to deal with shared parameters in the convolutional kernels.

In Table I, we compare the test accuracies obtained on MNIST-DVS with both models after the sequential presentation of $100,000$ training examples, each selected (with replacement) from the training set. With DECOLLE, examples are presented in *mini-batches* of size 100, while the GLM-based SNN processes one at a time in an online fashion. As seen in Table I, the SRM model with a convolutional architecture outperforms all other solutions, as long as one can accomodate the necessary number of hidden neurons and operate as a sufficiently high sampling rate. In contrast, in resource-constrained environments, when the sampling period and number of hidden neurons decreases, the performance of SRM-based training is significantly affected, while GLM-based solutions prove more robust. This suggests that the use of a global learning signal along with random sampling in GLM-based SNNs is less affected than local feedback signals by a reduction in model size and sampling rate.

Next, we compare the training convergence speed of both models. To this end, we train the SNNs on the DVSGesture dataset. We set $N_H = 256$ and consider the same systems described above, to which we add an SNN trained with DECOLLE by presenting training examples one by one, i.e., with mini-batches of size one. We report the evolution of the training accuracies after training over $25,000$ examples. We note that training an SRM with DECOLLE is generally faster than training a GLM model as long as a sufficiently large mini-batch size is used. This suggests that, in this case, local per-neuron error signals can provide more informative updates than global error signals driven by random sampling.

V. CONCLUSIONS

In this paper, we have reviewed the state of the art on the use of SNNs as processors of spatio-temporal information in spiking signals. To that end, we have described SNNs as RNNs, and derived algorithms to train SRM and GLM-based SNN models. Based on neuromorphic datasets, we have provided insights on the learning performance and convergence under both models.

REFERENCES

[1] B. Rueckauer, I.-A. Lungu, Y. Hu, M. Pfeiffer, and S.-C. Liu, "Conversion of continuous-valued deep networks to efficient event-driven networks for image classification," *Frontiers in Neuroscience*, vol. 11, p. 682, 2017.

[2] Y. Cao, Y. Chen, and D. Khosla, "Spiking deep convolutional neural networks for energy-efficient object recognition," *International Journal of Computer Vision*, vol. 113, no. 1, pp. 54–66, 2015.

[3] I. M. Comsa *et al.*, "Temporal coding in spiking neural networks with alpha synaptic function," in *Proc. of International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2020.

[4] M. Davies, "Benchmarks for progress in neuromorphic computing," *Nature Machine Intelligence*, vol. 1, no. 9, pp. 386–388, 2019.

[5] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 x 128 120db 30mw asynchronous vision sensor that responds to relative intensity change," in *Proc. of International Solid-State Circuits Conference*. IEEE, 2006.

[6] A. Amir *et al.*, "A low power, fully event-based gesture recognition system," in *Proc. of Conference on Computer Vision and Pattern Recognition*. IEEE, 2017.

[7] B. Cramer, Y. Stradmann, J. Schemmel, and F. Zenke, "Heidelberg spiking datasets," 2019. [Online]. Available: http://dx.doi.org/10.21227/51gn-m114

[8] E. O. Neftci, H. Mostafa, and F. Zenke, "Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 51–63, 2019.

[9] F. Zenke and S. Ganguli, "Superspike: Supervised learning in multilayer spiking neural networks," *Neural Computation*, vol. 30, no. 6, pp. 1514–1541, 2018.

[10] J. Kaiser, H. Mostafa, and E. Neftci, "Synaptic plasticity dynamics for deep continuous local learning (DECOLLE)," *Frontiers in Neuroscience*, vol. 14, p. 424, 2020.

[11] J. W. Pillow *et al.*, "Spatio-temporal correlations and visual signalling in a complete neuronal population," *Nature*, vol. 454, no. 7207, p. 995, 2008.

[12] W. Gerstner and W. M. Kistler, *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge University Press, 2002.

[13] H. Jang, O. Simeone, B. Gardner, and A. Grüning, "An introduction to probabilistic spiking neural networks: Probabilistic models, learning rules, and applications," *IEEE Signal Processing Magazine*, vol. 36, no. 6, pp. 64–77, 2019.

[14] S. Woźniak, A. Pantazi, T. Bohnstingl, and E. Eleftheriou, "Deep learning incorporating biologically inspired neural dynamics and in-memory computing," *Nature Machine Intelligence*, vol. 2, no. 6, pp. 325–336, 2020.

[15] S. M. Bohte, H. La Poutré, and J. N. Kok, "Unsupervised clustering with spiking neurons by sparse temporal coding and multilayer RBF networks," *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 426–435, 2002.

[16] G. Kramer, *Directed information for channels with feedback*. Hartung-Gorre, 1998.

[17] S. M. Bohte, J. N. Kok, and H. La Poutre, "Error-backpropagation in temporally encoded networks of spiking neurons," *Neurocomputing*, vol. 48, no. 1-4, pp. 17–37, 2002.

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

[19] S. B. Shrestha and G. Orchard, "Slayer: Spike layer error reassignment in time," in *Proc. of Advances in Neural Information Processing Systems*, 2018.

[20] N. Frémaux and W. Gerstner, "Neuromodulated spike-timing-dependent plasticity, and theory of three-factor learning rules," *Frontiers in Neural Circuits*, vol. 9, p. 85, 2016.

[21] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT Press, 2009.

[22] A. Mnih and K. Gregor, "Neural variational inference and learning in belief networks," in *Proc. of International Conference on Machine Learning*, 2014.

[23] K. T. Michalis and M. Lázaro-Gredilla, "Local expectation gradients for black box variational inference," in *Proc. of Advances in Neural Information Processing Systems*, 2015.

[24] T. Serrano-Gotarredona and B. Linares-Barranco, "Poker-DVS and MNIST-DVS: Their history, how they were made, and other details," *Frontiers in Neuroscience*, vol. 9, p. 481, 2015.