TRAINING DYNAMIC EXPONENTIAL FAMILY MODELS WITH CAUSAL AND LATERAL DEPENDENCIES FOR GENERALIZED NEUROMORPHIC COMPUTING

Hyeryung Jang and Osvaldo Simeone

Department of Informatics, King's College London, United Kingdom

ABSTRACT

Neuromorphic hardware platforms, such as Intel's Loihi chip, support the implementation of Spiking Neural Networks (SNNs) as an energy-efficient alternative to Artificial Neural Networks (ANNs). SNNs are networks of neurons with internal analogue dynamics that communicate by means of binary time series. In this work, a probabilistic model is introduced for a generalized set-up in which the synaptic time series can take values in an arbitrary alphabet and are characterized by both causal and instantaneous statistical dependencies. The model, which can be considered as an extension of exponential family harmoniums to time series, is introduced by means of a hybrid directed-undirected graphical representation. Furthermore, distributed learning rules are derived for Maximum Likelihood and Bayesian criteria under the assumption of fully observed time series in the training set.

Index Terms— Spiking Neural Network (SNN), exponential family model, Maximum Likelihood, Bayesian learning, neuromorphic computing

1. INTRODUCTION

The current dominant computing framework for supervised learning applications is given by feed-forward multi-layer Artificial Neural Networks (ANNs). These systems process real numbers through a cascade of non-linearities applied successively over multiple layers. It is well-understood that training and running ANNs for inference generally require a significant amount of resources in terms of space and time (see, e.g., [1]). Neuromorphic computing, currently backed by recent major projects by IBM, Qualcomm, and Intel, offers a fundamental paradigm shift that takes the trend towards distributed computing initiated by ANNs to its natural extreme by borrowing insights from computational neuroscience. A neuromorphic chip consists of a network of spiking neurons with internal temporal analogue dynamics and digital spike-based synaptic communications. Current hardware implementations confirm drastic power reductions by orders of magnitude with respect to ANNs [2, 3].

Models typically used to train Spiking Neural Networks (SNNs) are deterministic, and learning rules borrow tools and ideas from the design of ANNs. Examples include the standard leaky integrate-and-fire model and its variants, with associated learning rules that approximate backpropagation [4–6]. Probabilistic models for SNNs are more conventionally adopted in computational neuroscience and offer a variety of potential advantages, including flexibility and availability of principled learning criteria [7, 8]. Nevertheless, they pose technical challenges in the design of training and inference algorithms that have only partially been addressed [9–12].

This work was supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 725731).

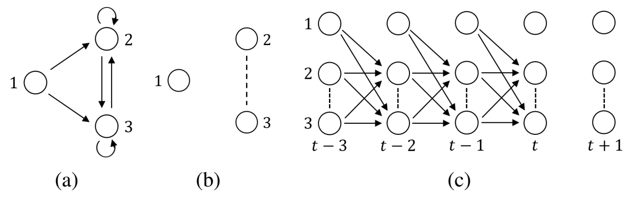

Fig. 1. Hybrid directed and undirected graphical representation of a dynamic exponential family model: (a) causal graph $\mathcal{G}_{\mathcal{P}}$ representing directed causal relationships; (b) lateral graph $\mathcal{G}_{\mathcal{L}}$ encoding instantaneous correlations; and (c) time-expanded graph assuming a memory for the causal connections equal to $\tau = 1$.

In this work, we study the problem of training a probabilistic model for correlated time series. The framework generalizes probabilistic models for SNNs [12, 13] by allowing for arbitrary alphabets – not restricted to binary – and by enabling individual time series – or neuron signals – not only to affect each other causally over time but also to have instantaneous correlations. It is noted that both general discrete alphabets and lateral dependencies are in principle implementable on existing digital neuromorphic chips [1]. We derive distributed learning rules for Maximum Likelihood (ML) and for Bayesian criteria based on synaptic sampling [14–18], and detail the corresponding communication requirements. Applications of the model to supervised learning via classification are also discussed.

The proposed model can be considered as an extension of exponential family harmoniums [19] from vector-valued signals to time series. We refer to the model as *dynamic exponential family*, which is specified by a hybrid directed and undirected graphical representation (see Fig. 1). We specifically focus here on the case of fully observed time series, which is of independent interest and also serves as building block for the more complex set-up with latent variables, to be considered in follow-up work.

The rest of this paper is organized as follows. In Sec. 2, we describe the dynamic exponential family model. Under this model, we then derive distributed learning rules for ML and Bayesian criteria in Sec. 3. Finally, Sec. 4 presents numerical results for a multi-task supervised learning problem tackled via a two-layer SNN.

2. DYNAMIC EXPONENTIAL FAMILY MODEL

In this section, we describe the proposed probabilistic model as an extension of exponential family harmoniums [19] to time series.

Hybrid directed-undirected representation. We consider a probabilistic model for dependent time series that captures both causal and lateral, or instantaneous, correlations. To describe both causal and lateral dependencies, we introduce a directed graph $\mathcal{G}_{\mathcal{P}} = (\mathcal{V}, \mathcal{E}_{\mathcal{P}})$ and an undirected graph $\mathcal{G}_{\mathcal{L}} = (\mathcal{V}, \mathcal{E}_{\mathcal{L}})$, respectively, where $\mathcal{V} = \{1, \ldots, N_x\}$ is the set of time sequences of interest. Having in mind the application to SNNs discussed above, we will also refer to the vertices $\mathcal{V}$ as neurons or units. The edge set $\mathcal{E}_{\mathcal{P}} \subseteq \mathcal{V}^2$ represents the *causal* connection between time series and corresponding neurons, and hence we refer to $\mathcal{G}_{\mathcal{P}}$ as the causal graph. Note that the presence of self-loops, *i.e.*, edges of the form (i, i) connecting a unit i to itself, or of more general loops involving multiple nodes, is allowed in the causal graph, and it indicates a recurrent behavior for the time series. In contrast, the edge set $\mathcal{E}_{\mathcal{L}} \subseteq \mathcal{V}^2$ encodes instantaneous correlations or *lateral* connections, and hence the graph $\mathcal{G}_{\mathcal{L}}$ is referred to as the lateral graph. Examples are shown in Fig. 1(a) and Fig. 1(b) for a system with $N_x = 3$ time series and thus $N_x = 3$ nodes.

In the causal graph $\mathcal{G}_{\mathcal{P}}$, we denote by $\mathcal{P}_i := \{j : (j,i) \in \mathcal{E}_{\mathcal{P}}\}$ the subset of units that has a causal connection to unit i. Note that the set $\mathcal{P}_i$ includes also unit i if there is a self-loop for unit i and that a causal connection $(j,i) \in \mathcal{E}_{\mathcal{P}}$ from unit j to unit i does not imply the inclusion of the edge (i,j) in $\mathcal{E}_{\mathcal{P}}$ in general. For the lateral graph $\mathcal{G}_{\mathcal{L}}$, we denote by $\mathcal{L}_i := \{j : (i,j) \in \mathcal{E}_{\mathcal{L}}\}$ the subset of units that has a lateral connection with unit i. Edges (i,j) and (j,i) are equivalent in graph $\mathcal{E}_{\mathcal{L}}$. Moreover, we denote by $\mathcal{C}_i$ the subset of units that are reachable from unit i in lateral graph $\mathcal{G}_{\mathcal{L}}$ in the sense that there exists a path in $\mathcal{G}_{\mathcal{L}}$ between units i and any unit $m \in \mathcal{C}_i$, where a path is a sequence of edges in $\mathcal{E}_{\mathcal{L}}$ that share a vertex. We generally have the inclusion $\mathcal{L}_i \subseteq \mathcal{C}_i$. In example of Fig. 1(b), we have the subsets $\mathcal{C}_1 = \emptyset$, $\mathcal{C}_2 = \{3\}$, and $\mathcal{C}_3 = \{2\}$.

Probabilistic model. Each unit $i \in \mathcal{V}$ is associated with a random sequence $\mathrm{x}_{i,t}$ for $t = 1, 2, \ldots$, taking values in either a continuous or discrete space $\mathcal{X}_i$. Given causal and lateral connections defined by graphs $\mathcal{G}_{\mathcal{P}}$ and $\mathcal{G}_{\mathcal{L}}$, the joint distribution of the random process $\mathbf{x}_t = (\mathrm{x}_{1,t}, \ldots, \mathrm{x}_{N_x,t})$ for $t = 1, 2, \ldots$ factorizes according to the chain rule

$$p_\Theta(\boldsymbol{x}^T) = \prod_{t=1}^{T} p_\Theta(\boldsymbol{x}_t | \boldsymbol{x}^{t-1}), \tag{1}$$

where we denote as $\mathbf{x}^t = (\mathbf{x}_1, \ldots, \mathbf{x}_t)$ the overall random process from time 1 to t, and as Θ the set of model parameters. In (1), we have implicitly conditioned on initial value $\boldsymbol{x}_0$, which is assumed to be fixed. Furthermore, the conditional probability $p_\Theta(\boldsymbol{x}_t|\boldsymbol{x}^{t-1})$ follows a distribution in the exponential family with N_a-dimensional sufficient statistics $\boldsymbol{s}_i(x_i) := [s_{i,1}(x_i), \ldots, s_{i,N_a}(x_i)]^\top$ for every unit i and a quadratic energy function. In the practically relevant case for neuromorphic computing where the alphabets $\mathcal{X}_i$ are discrete and finite, *i.e.*, $\mathcal{X}_i = \{0, 1, \ldots, C-1\}$ for some integer C, the sufficient statistics $\boldsymbol{s}_i$ are defined as the one-hot representation, i.e., $\boldsymbol{s}_i(x_i) = [1_{\{x_i=1\}}, \ldots, 1_{\{x_i=C-1\}}]^\top$, where 1_E is the indicator function for the event E. Note that we have $N_a = C - 1$. Furthermore, SNNs are characterized by $C = 2$, i.e., by a binary alphabet, and we have $\boldsymbol{s}_i(x_i) = x_i \in \{0, 1\}$.

Writing $\boldsymbol{s}_{i,t} = \boldsymbol{s}_i(x_{i,t})$ for simplicity of notation, we have

$$p_\Theta(\boldsymbol{x}_t|\boldsymbol{x}^{t-1}) \propto \exp\Bigg\{ \sum_{i \in \mathcal{V}} \Big(\boldsymbol{\theta}_i^\top \boldsymbol{s}_{i,t} + \sum_{\delta=1}^{\tau} \sum_{j \in \mathcal{P}_i} \boldsymbol{s}_{j,t-\delta}^\top \mathbf{W}_{j,i}^{[\delta]} \boldsymbol{s}_{i,t} + \sum_{j \in \mathcal{L}_i} \boldsymbol{s}_{j,t}^\top \mathbf{U}_{j,i} \boldsymbol{s}_{i,t} \Big) \Bigg\}, \tag{2}$$

for some memory parameter $\tau > 0$. The model (2) captures causal and lateral connections through the second and third terms of the exponential function, respectively. The model is parameterized by the set $\Theta = \{\boldsymbol{\theta}, \{\mathbf{W}^{[\delta]}\}_{\delta \in \{1,\ldots,\tau\}}, \mathbf{U}\}$, whose components are described next.

First, the vector $\boldsymbol{\theta}_i := [\theta_{i,1}, \ldots, \theta_{i,N_a}]^\top$ contains unit-wise natural parameter for unit i, which is collected as $\boldsymbol{\theta} = \{\boldsymbol{\theta}_i\}_{i \in \mathcal{V}}$. Second, for every pair of units $(j,i) \in \mathcal{E}_{\mathcal{P}}$, the $N_a \times N_a$ matrix $\mathbf{W}_{j,i}^{[\delta]}$ describes the causal influence of unit j to unit i after a time lag of $\delta \in \{1, \ldots, \tau\}$ time instants. We use the notations $\mathbf{W}^{[\delta]} = \{\mathbf{W}_{j,i}^{[\delta]}\}_{(j,i) \in \mathcal{E}_{\mathcal{P}}}$ and $\mathbf{W} = \{\mathbf{W}^{[\delta]}\}_{\delta \in \{1,\ldots,\tau\}}$. The structure of the causal graph $\mathcal{G}_{\mathcal{P}}$ defines the position of zeros in $\mathbf{W}$: $\mathbf{W}_{j,i}^{[\delta]}$ is generally non-zero if $(j,i) \in \mathcal{E}_{\mathcal{P}}$ and is an all-zero matrix otherwise. Third, for every pair of units $(j,i) \in \mathcal{E}_{\mathcal{L}}$, the $N_a \times N_a$ matrix $\mathbf{U}_{j,i}$ describes the instantaneous dependence between units j and i. The structure of the lateral graph $\mathcal{G}_{\mathcal{L}}$ defines the position of zeros in $\mathbf{U} = \{\mathbf{U}_{j,i}\}_{(j,i) \in \mathcal{E}_{\mathcal{L}}}$: $\mathbf{U}_{j,i}$ is generally non-zero if $(j,i) \in \mathcal{E}_{\mathcal{L}}$ and is an all-zero matrix otherwise. Note also that we have $\mathbf{U}_{j,i} = \mathbf{U}_{i,j}$ for $(j,i) \in \mathcal{E}_{\mathcal{L}}$.

Following the common approach in computational neuroscience [20], we adopt the parameterization of the causal parameters $\mathbf{W}$ as the weighted sum of fixed basis functions with learnable weights. To elaborate, for every $(j,i) \in \mathcal{E}_{\mathcal{P}}$, we write

$$\mathbf{W}_{j,i}^{[\delta]} = \sum_{k=1}^{K} a_k^{[\delta]} \mathbf{V}_{j,i,k}, \tag{3}$$

where we have defined K basis functions $a_k^{[\delta]}$, which vary over time $\delta \in \{1, \ldots, \tau\}$ for $k = 1, \ldots, K$, and learnable weight matrices $\mathbf{V}_{j,i} = [\mathbf{V}_{j,i,1}, \ldots, \mathbf{V}_{j,i,K}]$ with $\mathbf{V}_{j,i,k}$ being a $N_a \times N_a$ matrix, which are collected as $\mathbf{V} = \{\mathbf{V}_{j,i}\}_{(j,i) \in \mathcal{E}_{\mathcal{P}}}$. We note that a set of basis functions, along with the weight matrices, describes the spatio-temporal receptive field of the neurons [20]. Examples of basis functions include raised cosines with different synaptic delays [20–22]. As a result, the model parameters are defined as $\Theta = \{\boldsymbol{\theta}, \mathbf{V}, \mathbf{U}\}$.

Plugging (3) into (2), it is observed that the second term of the exponential function can be expressed in terms of the k-th *filtered trace* of unit j, which we define as the convolution

$$\boldsymbol{\alpha}_{j,k,t} := \sum_{\delta=0}^{\tau-1} a_k^{[\delta+1]} \boldsymbol{s}_{j,t-\delta}. \tag{4}$$

This signal is obtained by filtering the signal $\boldsymbol{s}_{j,t}$ through a filter with impulse response equal to the basis function $a_k^{[\delta]}$. The set of filtered traces $\boldsymbol{\alpha}_{j,t-1} = [\boldsymbol{\alpha}_{j,1,t-1}, \ldots, \boldsymbol{\alpha}_{j,K,t-1}]$ from all units j in the set $\mathcal{P}_i$ determines the *generalized membrane potential* of unit i at time t

$$\boldsymbol{r}_{i,t} = \boldsymbol{\theta}_i + \sum_{j \in \mathcal{P}_i} \sum_{k=1}^{K} \mathbf{V}_{j,i,k}^\top \boldsymbol{\alpha}_{j,k,t-1}. \tag{5}$$

With the help of this definition, we can rewrite the conditional joint distribution in (2) as

$$p_\Theta(\boldsymbol{x}_t|\boldsymbol{x}^{t-1}) = p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t) \tag{6a}$$

$$\propto \exp\Big\{ \sum_{i \in \mathcal{V}} \boldsymbol{r}_{i,t}^\top \boldsymbol{s}_{i,t} + \sum_{(j,i) \in \mathcal{E}_{\mathcal{L}}} \boldsymbol{s}_{j,t}^\top \mathbf{U}_{j,i} \boldsymbol{s}_{i,t} \Big\}. \tag{6b}$$

In (6a), we have emphasized the dependence of $\boldsymbol{x}_t$ on the history $\boldsymbol{x}^{t-1}$ only through the membrane potentials $\boldsymbol{r}_t = \{\boldsymbol{r}_{i,t}\}_{i \in \mathcal{V}}$. We refer to technical report [23] for further discussion.

3. LEARNING

In this section, we tackle the problem of learning the model parameter Θ for fixed basis functions under the assumption of fully

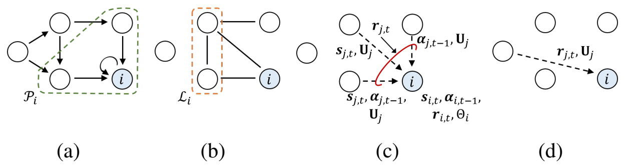
$\mathcal{P}_i$
i
$\mathcal{L}_i$
$\boldsymbol{r}_{j,t}$
$\boldsymbol{s}_{j,t}, \mathbf{U}_j$
$\boldsymbol{\alpha}_{j,t-1}, \mathbf{U}_j$
$\boldsymbol{s}_{j,t}, \boldsymbol{\alpha}_{j,t-1}, \mathbf{U}_j$
$\boldsymbol{s}_{i,t}, \boldsymbol{\alpha}_{i,t-1}, \boldsymbol{r}_{i,t}, \Theta_i$
$\boldsymbol{r}_{j,t}, \mathbf{U}_j$
(a)
(b)
(c)
(d)

Fig. 2. Illustration of the information required to compute the gradients (7) for unit i: (a) directed graph $\mathcal{G}_{\mathcal{P}}$; (b) undirected graph $\mathcal{G}_{\mathcal{L}}$; (c) local and (d) non-local information necessary for learning.

observed time series. As we will see in Sec. 4, this set-up is, for instance, relevant to the training of two-layer SNNs and generalizations thereof, in supervised learning problems.

3.1. Maximum Likelihood Learning via Gradient Descent

We first aim at deriving gradient-based rules for ML training. To this end, we write the gradient of the likelihood function $\mathcal{L}_{\boldsymbol{x}^T}(\Theta) := \ln p_\Theta(\boldsymbol{x}^T)$ as $\nabla_\Theta \mathcal{L}_{\boldsymbol{x}^T}(\Theta) = \sum_{t=1}^T \nabla_\Theta \ln p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t)$, where

$$\nabla_{\boldsymbol{\theta}_i} \ln p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t) = \boldsymbol{s}_{i,t} - \mathbb{E}_{\mathbf{x}_t \sim p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t)}\big[\mathbf{s}_{i,t}|\boldsymbol{r}_t\big], \tag{7a}$$

$$\nabla_{\mathbf{V}_{j,i}} \ln p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t) = \boldsymbol{\alpha}_{j,t-1}\big(\boldsymbol{s}_{i,t} - \mathbb{E}_{\mathbf{x}_t \sim p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t)}\big[\mathbf{s}_{i,t}|\boldsymbol{r}_t\big]\big)^\top, \tag{7b}$$

$$\nabla_{\mathbf{U}_{j,i}} \ln p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t) = \boldsymbol{s}_{j,t}\boldsymbol{s}_{i,t}^\top - \mathbb{E}_{\mathbf{x}_t \sim p_\Theta(\boldsymbol{x}_t|\boldsymbol{r}_t)}\big[\mathbf{s}_{j,t}\mathbf{s}_{i,t}^\top|\boldsymbol{r}_t\big]. \tag{7c}$$

The gradients (7) have the structure, typical for exponential family models, that presents the difference between the empirical average of the relevant sufficient statistics given the actual realization of the time series $\boldsymbol{x}^T$ and the corresponding ensemble average under the model distribution (6) (see, e.g., [24]). A stochastic gradient-based optimizer iteratively updates the parameter Θ based on the rule

$$\Theta \leftarrow \Theta + \eta \nabla_\Theta \mathcal{L}_{\boldsymbol{x}^T}(\Theta), \tag{8}$$

by drawing an example $\boldsymbol{x}^T$ from the training set, where the learning rate η is assumed to be fixed here for simplicity of notation. With this rule, each unit $i \in \mathcal{V}$ updates its own local parameters $\Theta_i = \{\boldsymbol{\theta}_i, \mathbf{V}_i, \mathbf{U}_i\}$, where we use the notations $\mathbf{V}_i = \{\mathbf{V}_{j,i}\}_{j\in\mathcal{P}_i}$ and $\mathbf{U}_i = \{\mathbf{U}_{j,i}\}_{j\in\mathcal{L}_i}$.

Distributed computation of the gradients. We now discuss the information required to compute the gradients (7) for each unit i in order to enable a distributed implementation of the rule (8). We define as *local* to neuron i all quantities that concern units that have either a directed edge to neuron i in $\mathcal{G}_{\mathcal{P}}$ or an undirected edge with it in $\mathcal{G}_{\mathcal{L}}$. The first terms of all the gradients (7) require the local statistics $\boldsymbol{s}_{i,t} = \boldsymbol{s}_i(x_{i,t})$ and the local filtered traces $\boldsymbol{\alpha}_{j,t-1}$ for all parents $j \in \mathcal{P}_i$. Furthermore, they also require the current local sufficient statistics $\boldsymbol{s}_{j,t}$ for all units $j \in \mathcal{L}_i$ connected by lateral edge. In contrast, the second terms entail the computation of the average of the local sufficient statistics $\mathbf{s}_{i,t}$ and of the products $\mathbf{s}_{j,t}\mathbf{s}_{i,t}^\top$ for $j \in \mathcal{L}_i$ over their marginal distributions, which is further discussed next.

The marginal distribution of the local sufficient statistics $\mathbf{s}_{i,t}$ is obtained from $p_\Theta(x_{i,t}|\boldsymbol{r}_t)$

$$p_\Theta(x_{i,t}|\boldsymbol{r}_t) = \sum_{\boldsymbol{x}_{\mathcal{C}_i,t}} p_{\Theta_i \cup \mathbf{U}_{\mathcal{C}_i}}(x_{i,t}, \boldsymbol{x}_{\mathcal{C}_i,t}|\boldsymbol{r}_{\mathcal{C}_i\cup\{i\},t}), \tag{9}$$

where we recall that $\mathcal{C}_i$ is the subset of units that are reachable from unit i via lateral connections. As indicated by the notation in (9), the joint distribution on the right-hand side can be obtained from (6b) by including in the sums at the exponent only the terms that depend on $\boldsymbol{s}_{i,t}$ and $\boldsymbol{s}_{\mathcal{C}_i,t}$. Calculation of (9) hence requires knowledge of the membrane potentials from the units in $\mathcal{C}_i \cup \{i\}$, and it depends on the local parameters Θ_i and lateral parameters $\mathbf{U}_{\mathcal{C}_i} = \{\mathbf{U}_j\}_{j\in\mathcal{C}_i}$ of all units j in the set $\mathcal{C}_i$. The marginal distribution of the products $\mathbf{s}_{j,t}\mathbf{s}_{i,t}^\top$ for $(j,i) \in \mathcal{E}_{\mathcal{L}}$ is similarly computed as

$$p_\Theta(x_{j,t}, x_{i,t}|\boldsymbol{r}_t) = \sum_{\boldsymbol{x}_{\mathcal{C}_i\setminus\{j\},t}} p_{\Theta_i \cup \mathbf{U}_{\mathcal{C}_i}}(x_{j,t}, x_{i,t}, \boldsymbol{x}_{\mathcal{C}_i\setminus\{j\},t}|\boldsymbol{r}_{\mathcal{C}_i\cup\{i\},t}). \tag{10}$$

In summary, as illustrated in Fig. 2, the computation of the gradients (7) for unit i requires knowledge of the local sufficient statistics $\boldsymbol{s}_{i,t}$ and $\boldsymbol{s}_{j,t}$ for all $j \in \mathcal{L}_i$, of the local filtered traces $\boldsymbol{\alpha}_{i,t-1}$ and $\boldsymbol{\alpha}_{j,t-1}$ for all $j \in \mathcal{P}_i$, as well as the local membrane potentials $\boldsymbol{r}_{j,t}$ and the current lateral parameters $\mathbf{U}_j$ for all $j \in (\mathcal{L}_i \cup \mathcal{P}_i) \cap \mathcal{C}_i$. It also requires the acquisition of the membrane potentials $\boldsymbol{r}_{j,t}$ and of the current lateral parameters $\mathbf{U}_j$ for all units j in the set $\mathcal{C}_i$. This calls for non-local signaling if there are units in $\mathcal{C}_i$ that are not in the sets $\mathcal{L}_i$ or $\mathcal{P}_i$.

The complexity of the computation of the second terms in (7) depends exponentially on the size of the set $\mathcal{C}_i$. In particular, if there are no lateral connections, *i.e.*, $\mathcal{G}_{\mathcal{L}}$ is an empty graph, the marginalizations (9)-(10) are not required since the set $\mathcal{C}_i$ of every unit i is empty, and the second terms can be obtained in closed form. When the sets $\mathcal{C}_i$ are too large to enable computation of the sums in (9)-(10), Gibbs sampling, or approximations such as Contrastive Divergence, can be used to enable the expectation [25].

3.2. Bayesian Learning

As a potentially more powerful alternative to ML learning, Bayesian methods aim at computing the posterior distribution of the parameter vector Θ under the assumption of a given prior $p(\Theta)$. Bayesian learning can capture parameter uncertainty and is able to naturally avoid overfitting. While exact Bayesian learning is prohibitively complex, a gradient-based method has been recently proposed that combines Robbins-Monro type stochastic approximation methods [26] with Langevin dynamics [14]. The method, called the *stochastic gradient Langevin dynamics* [15–18], updates the parameter as

$$\Theta \leftarrow \Theta + \eta\bigg(\nabla_\Theta \ln p(\Theta) + N\nabla_\Theta \mathcal{L}_{\boldsymbol{x}^T}(\Theta)\bigg) + \sqrt{2\eta}\boldsymbol{\nu}, \tag{11}$$

by drawing an example $\boldsymbol{x}^T$ from the training set size of N, where η is the learning rate and $\boldsymbol{\nu}$ denotes i.i.d. Gaussian noise with zero mean and variance 1. The update rule (11) has the property that the distribution of the parameter vector Θ at equilibrium matches the Bayesian posterior as $\eta \to 0$.

4. RESULTS AND DISCUSSION

In this section, we consider an application of the methods developed in this paper to a *multi-task* supervised learning problem based on a two-layer SNN. The problem consists of the two tasks of classifying a handwritten number and of detecting a possible rotation of the image of the handwritten digit (see Fig. 3). As illustrated in Fig. 3, the set of neurons $\mathcal{V}$ consists of two layers: the spike trains associated with the neurons in the first layer are determined by the input image, while the spike trains corresponding to the second layer are

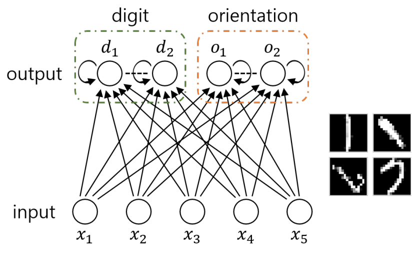

Fig. 3. Hybrid directed and undirected graphical architecture of a two-layer SNN used for supervised learning: each input neuron x_i corresponds to a pixel of the input image; each output neuron d_i encodes the digit index; and each output neuron o_i encodes the orientation index. The directed graph has causal connections from input neurons to output neurons and self-loops at the output neurons; and the undirected graph has lateral connections only between output neurons in the same subset.

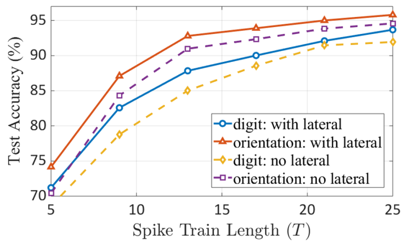

Fig. 4. ML learning: test accuracy for the two tasks of digit and orientation classification versus the duration T of the time series when $K = 2$.

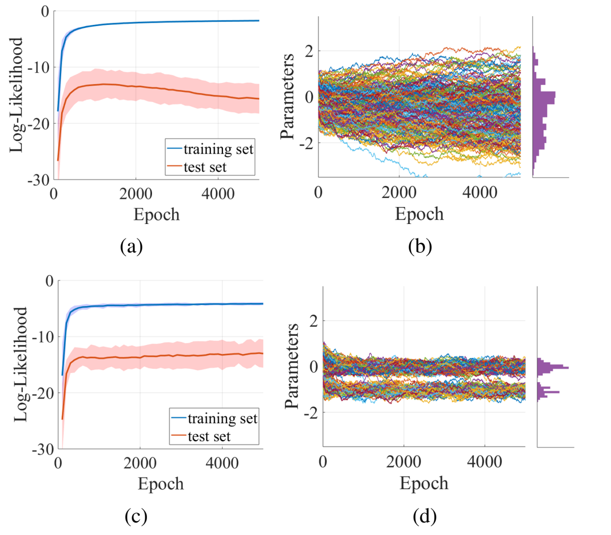

Fig. 5. Bayesian learning with different priors: (a)-(b) with uniform prior and (c)-(d) with bimodal prior; figures (a) and (c) show the log-likelihood on the training and test sets as the function of the learning epochs, where the shaded area indicates three standard deviations of the mean based on 20 trials; and figures (b) and (d) plot the evolution of 80 randomly selected causal parameters throughout the epochs (left) and the histogram at the end of the learning epoch (right).

determined by the correct labels for the indices of number and orientation of the image. Specifically, we have one input neuron for every pixel of the image, and the output layer contains two subsets of neurons: one subset of neurons with one neuron for every possible digit index, and another subset with one neuron for each possible orientation, i.e., vertical (unrotated) or rotated. The two-layer SNN can be represented as a dynamic exponential family, in which there are causal connections only from input neurons to output neurons, with self-loops at the output neurons, as well as lateral connections between output neurons in the same subset in order to capture the correlations of the labels in the same task.

The training dataset is generated by selecting 500 images of each digit "1" and "7" from the USPS handwritten digit dataset [27]. All 16×16 images are included as they are, that is, unrotated, and they are also included upon a rotation by a randomly selected degree. The test set is similarly obtained by using 125 examples from the USPS dataset. As a result, we have 256 input neurons, with one per pixel of the input image. Each gray scale pixel is converted to an input spike train by generating an i.i.d. Bernoulli vector of T samples with spike probability proportional to the pixel intensity and limited between 0 and 0.5. The digit labels $\{1, 7\}$ are encoded by the first subset of two output neurons, so that the desired output of the neuron corresponding to the correct label index is one spike emission every three non-spike samples, while the other neuron is silent. The same is done for the orientation labels $\{\mathtt{v}, \mathtt{r}\}$, using the second subset of two output neurons, where the label $\mathtt{v}$ indicates the original unrotated image, and the label $\mathtt{r}$ indicates the rotated image. In order to test the classification accuracy, we assume standard rate decoding, which selects the output neuron, and hence the corresponding indices, with the largest number of output spikes. Finally, in order to investigate the impact of lateral connections in our model, we also consider a two-layer SNN in which lateral connections are not allowed.

First, we train both models with the ML learning rule (8) for 800 epochs with constant learning rate $\eta = 0.05$, based on 20 trials with different random seeds. The model parameters are randomly initialized with uniform distribution in range $[-1, 1]$, while the lateral parameters are randomly initialized in range $[-2, 2]$. We assume the K raised cosine basis functions introduced in [9,20] in (3). Fig. 4 demonstrates the classification accuracy in the test set versus the length T of time series with $K = 2$. From the figure, the task of classifying orientation is seen to be easier than digit identification. Furthermore, we observe better accuracies in both tasks when we train the model with lateral connections between output neurons. This implies that the learning rule (8) can efficiently learn how to make use of the instantaneous correlations among output neurons participating in the same task.

We then consider the Bayesian learning rule (11), and study the impact of the prior $p(\Theta)$ for the model parameters on overfitting. To this end, we select 10 and 50 samples of each digit and orientation class for training and testing, respectively. After training the model for a number of epochs with constant learning rate $\eta = 0.000625$, we evaluate the learned model by measuring the log-likelihood of the desired output spikes for the correct label given the input images in both training and test set. For the experiment in Fig. 5(a)-5(b), a uniform prior over the causal parameters $\mathbf{W}$ is assumed; while we choose a mixture of two Gaussians with means 0 and -1.0 and identical standard deviations 0.15 as a prior for the results in Fig. 5(c)-5(d) (see [23] for details of the simulations). Fig. 5(a) shows that a uniform prior can lead to overfitting, as also suggested by the larger variance of the sampled weights plotted in Fig. 5(b). In contrast, a bimodal prior yields good generalization performance due to its capacity to act as a regularizer by keeping a fraction of the weights close to zero as seen in Fig. 5(d).

5. REFERENCES

[1] Mike Davies, Narayan Srinivasa, Tsung-Han Lin, Gautham Chinya, Yongqiang Cao, Sri Harsha Choday, Georgios Dimou, Prasad Joshi, Nabil Imam, Shweta Jain, et al., "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[2] Alan Diamond, Thomas Nowotny, and Michael Schmuker, "Comparing neuromorphic solutions in action: implementing a bio-inspired solution to a benchmark classification task on three parallel-computing platforms," *Frontiers in Neuroscience*, vol. 9, pp. 491, 2016.

[3] Catherine D Schuman, Thomas E Potok, Robert M Patton, J Douglas Birdwell, Mark E Dean, Garrett S Rose, and James S Plank, "A survey of neuromorphic computing and neural networks in hardware," *arXiv preprint arXiv:1705.06963*, 2017.

[4] Jun Haeng Lee, Tobi Delbruck, and Michael Pfeiffer, "Training deep spiking neural networks using backpropagation," *Frontiers in Neuroscience*, vol. 10, pp. 508, 2016.

[5] Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi, "Spatio-temporal backpropagation for training high-performance spiking neural networks," *Frontiers in Neuroscience*, vol. 12, 2018.

[6] Sander M Bohte, Joost N Kok, and Johannes A La Poutré, "Spikeprop: backpropagation for networks of spiking neurons.," in *ESANN*, 2000, pp. 419–424.

[7] Jonathan W Pillow, Liam Paninski, Valerie J Uzzell, Eero P Simoncelli, and EJ Chichilnisky, "Prediction and decoding of retinal ganglion cell responses with a probabilistic spiking model," *Journal of Neuroscience*, vol. 25, no. 47, pp. 11003–11013, 2005.

[8] Daphne Koller, Nir Friedman, and Francis Bach, *Probabilistic graphical models: principles and techniques*, MIT press, 2009.

[9] Alireza Bagheri, Osvaldo Simeone, and Bipin Rajendran, "Training probabilistic spiking neural networks with first-to-spike decoding," *IEEE ICASSP*, 2018.

[10] Brian Gardner and André Grüning, "Supervised learning in spiking neural networks for precise temporal encoding," *PLoS ONE*, vol. 11, no. 8, pp. e0161335, 2016.

[11] Jean-Pascal Pfister, Taro Toyoizumi, David Barber, and Wulfram Gerstner, "Optimal spike-timing-dependent plasticity for precise action potential firing in supervised learning," *Neural Computation*, vol. 18, no. 6, pp. 1318–1348, 2006.

[12] Danilo J Rezende and Wulfram Gerstner, "Stochastic variational learning in recurrent spiking networks," *Frontiers in Computational Neuroscience*, vol. 8, pp. 38, 2014.

[13] Geoffrey E Hinton and Andrew D Brown, "Spiking boltzmann machines," in *Advances in Neural Information Processing Systems*, 2000, pp. 122–128.

[14] Radford M Neal et al., "MCMC using Hamiltonian dynamics," *Handbook of Markov Chain Monte Carlo*, vol. 2, no. 11, pp. 2, 2011.

[15] Max Welling and Yee W Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *International Conference on Machine Learning*, 2011, pp. 681–688.

[16] Sam Patterson and Yee Whye Teh, "Stochastic gradient Riemannian Langevin dynamics on the probability simplex," in *Advances in Neural Information Processing Systems*, 2013, pp. 3102–3110.

[17] Issei Sato and Hiroshi Nakagawa, "Approximation analysis of stochastic gradient Langevin dynamics by using fokker-planck equation and ito process," in *International Conference on Machine Learning*, 2014, pp. 982–990.

[18] David Kappel, Stefan Habenschuss, Robert Legenstein, and Wolfgang Maass, "Network plasticity as Bayesian inference," *PLoS Computational Biology*, vol. 11, no. 11, pp. e1004485, 2015.

[19] Max Welling, Michal Rosen-Zvi, and Geoffrey E Hinton, "Exponential family harmoniums with an application to information retrieval," in *Advances in Neural Information Processing Systems*, 2005, pp. 1481–1488.

[20] Jonathan W Pillow, Jonathon Shlens, Liam Paninski, Alexander Sher, Alan M Litke, EJ Chichilnisky, and Eero P Simoncelli, "Spatio-temporal correlations and visual signalling in a complete neuronal population," *Nature*, vol. 454, no. 7207, pp. 995, 2008.

[21] Pierre Baldi and Amir F Atiya, "How delays affect neural dynamics and learning," *IEEE Transactions on Neural Networks*, vol. 5, no. 4, pp. 612–621, 1994.

[22] Aboozar Taherkhani, Ammar Belatreche, Yuhua Li, and Liam P Maguire, "Dl-resume: a delay learning-based remote supervised method for spiking neurons," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3137–3149, 2015.

[23] Hyeryung Jang and Osvaldo Simeone, "Training dynamic exponential family models with causal and lateral dependencies for generalized neuromorphic computing," *arXiv preprint arXiv:1810.08940*, 2018.

[24] Osvaldo Simeone et al., "A brief introduction to machine learning for engineers," *Foundations and Trends® in Signal Processing*, vol. 12, no. 3-4, pp. 200–431, 2018.

[25] Geoffrey E Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771–1800, 2002.

[26] Herbert Robbins and Sutton Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, vol. 22, no. 3, pp. 400–407, 1951.

[27] Jonathan J. Hull, "A database for handwritten text recognition research," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 5, pp. 550–554, 1994.