Minimal Spiking Neuron for Solving Multilabel Classification Tasks

Jakub Fil
J.Fil@kent.ac.uk
Dominique Chu
D.F.Chu@kent.ac.uk
School of Computing, University of Kent, Canterbury CT2 7NF, U.K.

The multispike tempotron (MST) is a powersul, single spiking neuron model that can solve complex supervised classification tasks. It is also internally complex, computationally expensive to evaluate, and unsuitable for neuromorphic hardware. Here we aim to understand whether it is possible to simplify the MST model while retaining its ability to learn and process information. To this end, we introduce a family of generalized neuron models (GNMs) that are a special case of the spike response model and much simpler and cheaper to simulate than the MST. We find that over a wide range of parameters, the GNM can learn at least as well as the MST does.

We identify the temporal autocorrelation of the membrane potential as the most important ingredient of the GNM that enables it to classify multiple spatiotemporal patterns. We also interpret the GNM as a chemical system, thus conceptually bridging computation by neural networks with molecular information processing. We conclude the letter by proposing alternative training approaches for the GNM, including error trace learning and error backpropagation.

1 Introduction

Spiking neurons have been shown to be computationally more powerful than standard rate-coded neurons (Maass, 1997; Gütig & Sompolinsky, 2006; Rubin, Monasson, & Sompolinsky, 2010). This motivates the hope that networks of spiking neurons (SNNs) can be built smaller than corresponding networks of rate-coded neurons while achieving the same computational task. This would be interesting because current deep architecture, while solving sophisticated tasks, also requires extremely large models. Networks requiring up to a billion of weights are not uncommon—for example, in language processing (Radford et al., 2018) or image recognition (Mahajan et al., 2018).

In practice, there are difficulties, however. Spiking neurons are more expensive to simulate than nonspiking units, which may outweigh any gains from the reduced network size. In some cases, this problem may be circumvented by using specialized neuromorphic hardware (Indiveri et al., 2011; Plana et al., 2011; Lin et al., 2018; Shahsavari, Devienne, & Boulet, 2019; Rajendran & Alibart, 2016) to model SNNs. However, this is not always practical or possible (Wunderlich et al., 2019).

In most cases, it will remain necessary to simulate SNNs on general-purpose computers. In order to be able to build efficient SNNs, we therefore need to understand systematically the computational properties of spiking neurons. Indeed, there exists a large number of variants of spiking models vastly varying in internal complexity (Hodgkin & Huxley, 1952; Izhikevich, 2003; Brunel & van Rossum, 2007; Gerstner, Kistler, Naud, & Paninski, 2014). These various neuronal models are not always developed with computational efficiency as a criterion. Particularly in computational neuroscience, considerations of (the rather vague concept of) biological plausibility are often more important than pure simplicity. Yet in the context of applications of spiking neurons in artificial intelligence (AI), biological plausibility is irrelevant, and the important criteria should be computational cost, ease of implementation, suitability for neuromorphic hardware, and performance. In order to be able to produce maximally parsimonious models, it is essential to first understand the necessary ingredients that enable computation. This will be the focus of this letter.

Concretely, here we investigate the minimal ingredients required for a single neuron to perform a multilabel classification task. The purpose of this task is to classify incoming spatiotemporal patterns into different classes and distinguish them from noise. In the SNN literature, there have been a number of attempts to solve variants of this task, including the remote supervised method (ReSuMe; Ponulak, 2005), the chronotron learning rule (Florian, 2012), or the spike pattern association neuron (Mohemmed, Schliebs, Matsuda, & Kasabov, 2012), and the precise spike-driven synaptic plasticity (Yu, Tang, Tan, & Li, 2013). For this letter, we focus on one of the most recent approaches: the multispike tempotron (MST; Gütig, 2016). This is a single neuron architecture that can be trained to distinguish fixed spatiotemporal patterns from (statistically indistinguishable) noise. Crucially, the MST can also classify patterns into different classes, where the class label is indicated by the number of output spikes released during the duration of the pattern. Noise is always considered as class 0—that is, the MST should not spike when presented with noise.

The neural dynamics of the MST can be summarized as follows: (1) The state of the neuron is defined by the value of the internal membrane potential $V(t)$. It is updated in discrete time. Spikes are generated when $V(t)$ crosses a set threshold value from below. (2) The inputs to the MST neuron are N (unmodelled) presynaptic neurons, spiking with a set frequency. (3) The MST does not accept directly spiking input, but it includes a preprocessing step whereby input spikes are converted into analog signals via a biexponential synapse. This could be interpreted as an additional layer of trivial, capacitor-like buffer neurons. (4) The membrane potential update rule of the MST takes into account the total spiking history of the presynaptic neurons. As a result, the future evolution of the neuron does not solely depend on the current state and inputs; it also depends on how the current state was reached. The update function also includes a constant decay of the membrane potential and a "soft" exponential reset following a spike. Unlike, for example, the leaky integrate-and-fire (LIF) neuron, the MST does not have an immediate hard reset to the resting potential or a refractory period.

In summary, while the MST is a powerful neuronal model, it is also internally complex. The question we address here is whether this internal complexity is necessary for the ability of the MST to learn. We find that it is not. Much simpler models can learn at least equally well. To show this, we systematically strip away features from the MST and check whether this has an impact on its ability to learn. To this end, we propose a family of generalized neuronal models (GNM), which is a special case of the well-known spike-response model (Gerstner & Kistler, 2002a; Jolivet, Lewis, & Gerstner, 2003). The GNM contains a number of readily interpretable parameters that we vary systematically in order to explore putatively crucial features of the model. Depending on how the parameters are set, we can approximate the MST model or implement radically simpler models. The most important parameters that we shall find are the spikiness of the GNM and its memory.

Using a rigorous exploration of the GNM parameter space, we find that most of the complexities of the MST neuron are not essential for learning. Indeed, there is no strict need for spiking, and is the soft reset is not important. However, we do find that a balanced amount of memory of past states—that is, a degree of temporal autocorrelation of the membrane potential—is crucial for learning. Interestingly, we identify the hard reset of the well-known LIF neuron (Gerstner et al., 2014), which erases any memory of prespike states, destroying the correlation of postspike and prespike membrane potentials, as a hindrance to learning in the single neuron model.

Following common practice in SNN, we assume that the GNM is updated in discrete time. However, we also find that a continuous time version of the GNM can classify well. This continuous time version is theoretically interesting because it lends itself to an interpretation as a chemical reaction network (CRN). We conclude our letter by showing how a very simple chemical system can be trained and used to perform multilabel classification.

2 Methods

2.1 The Generalized Neuron Model. The GNM is a parametrized family of models that can be tuned to display varying degrees of spikiness, temporal autocorrelation of the membrane potential, and hysteresis (see

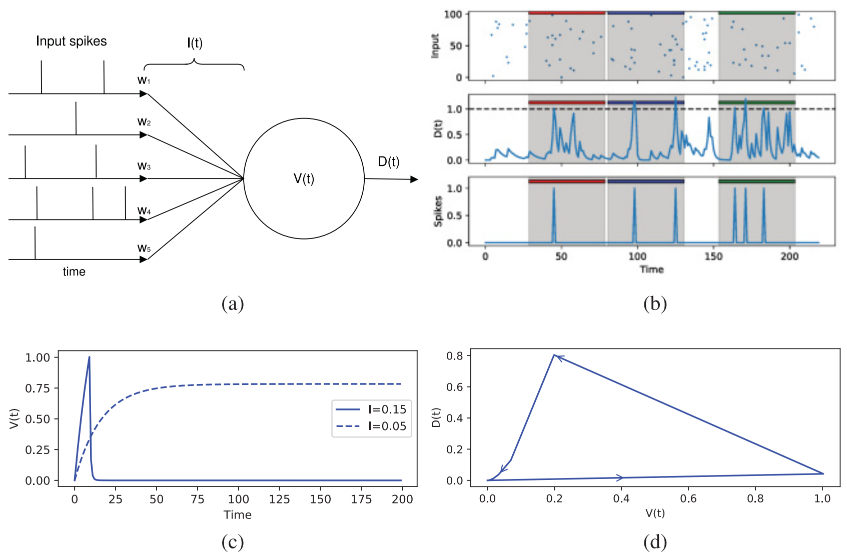


Figure 1: (a) Schematic outline of the GNM model. GNM has N weighted input connections, each of which receives temporal patterns of spikes. (b) When the readout function reaches a threshold ϑ_{R}, an output spike is recorded. In this example, three patterns are presented: red, blue, and green. The GNM responds with a single spike to the red pattern, two spikes to the blue pattern, and three to the green pattern. It should remain silent otherwise during a noisy phase. (c) Membrane potential as a function of time of the GNM stimulated by subthreshold (dashed line) and superthreshold (solid line) continuous input, with parameters $\eta = 0.8$, $\alpha = 0.3$, $\beta = 0.1$. (d) Hysteretic behavior of $D(t)$ as a function of $V(t)$ in the presence of superthreshold input.

Figure 1). The model is defined by the update function of the membrane potential $V(t)$. If updates are made in discrete time, as is usual in the SNN literature, then the model is as follows:

$$V(t) - V(t-1) = I(t) - \underbrace{(\eta\gamma R(t-1)V(t-1) + (1-\eta)\alpha V(t-1))}_{:=D(t)}$$

$$R(t) - R(t-1) = \zeta \frac{V(t-1)^h}{\vartheta_{\mathrm{B}}^h + V(t-1)^h} - \beta R(t-1). \tag{2.1a}$$

Here, $I(t) := \sum_{i=1}^{M} w_i \delta(t_j^i - t)$ is the sum of weighted inputs at time t, and t_j^i is the time of the jth spike of input i, where i runs from 1 to M; $\delta(x)$ is 1 if $x = 0$ and 0 otherwise; $0 \leq w_i \leq 1$ is the weight of the ith input. α and γ are decay coefficients of the membrane potential, ϑ_{B} is a behavioral threshold, and $0 \leq$

$\eta \leq 1$ is a model choice parameter. ζ is a rate parameter of the Hill function, h is a Hill function coefficient, and β defines the decay rate of R. The model is best understood by considering some special parameter choices.

For $\eta = 0$, the update function of the membrane potential reduces to a leaky integrator:

$$V(t) - V(t-1) = I(t) - \alpha V(t-1). \tag{2.1b}$$

In this case, the membrane potential is increased by whatever the input I is at time t, and it decays by a constant factor α. In the limiting case of $\alpha = 0$, there is no decay at all. The opposite extreme is $\alpha = 1$, which means that at each time step all the previous membrane potential is forgotten. For the discrete model it is not meaningful to set $\alpha > 1$ or $\alpha < 0$. The parameter α determines the temporal autocorrelation of the membrane potential, or the "memory." We will find this to be a crucial parameter for the performance of the GNM.

When $\eta > 0$, an additional time-dependent decay rate R becomes relevant. R can be thought of as describing the number of ion channels that open only after the membrane potential approaches the threshold ϑ_{B} and close, stochastically, with a rate β. Alternatively, this can be understood as the simplest model that implements a soft postspike reset.

At the start of a simulation, $R(0)$ will be set to 0. The subsequent increase of $R(t)$ depends on the membrane potential via a Hill-function (the first term on the right-hand side of equation 2.1a), which is a sigmoidal activation function. As $h \to \infty$, the Hill function approaches a step–function with a transition at the point $V = \vartheta_{\mathrm{B}}$. Even for finite values of h, the Hill function will be close to zero (one) when the membrane potential $V(t)$ is below (above) ϑ_{B}. Note that the decay rate R decays itself with a rate of β.

The effect of this additional decay mode is that the membrane potential may decay faster after having crossed a threshold value ϑ_{B}. This introduces a memory about past spike events into the model. The duration of the reset depends on the value of β and continues even if the membrane potential falls back below the threshold. Thus, the model has hysteresis (see Figure 1d).

Before continuing, it is useful to discuss briefly the relation between the GNM model and other well-known neuronal models. Unlike the MST, the update rule of the GNM is purely state dependent. The η parameter can be seen as regulating the "spikiness" of the model. The soft reset of the MST can be simulated in the GNM when $\eta > 0$ and the values of the parameters β, ζ are set appropriately. The well-known LIF neuron behaves like the GNM model with $h = \infty$, $\zeta = 1$, $\gamma = 1/\eta$, and $0 < \eta < 1$, up to reaching the behavioral threshold ϑ_{B}, including the postspike reset. However, following the reset, the LIF undergoes a deterministic refractory period, during which it remains insensitive to inputs. This type of fixed refractory period cannot be simulated by the GNM.

In addition to the discrete time dynamics of equation 2.1a, we also show simulations of the full continuous time dynamics. In order to describe the continuous dynamics, we need to convert the difference equations, equation 2.1a into proper differential equations, thus obtaining:

$$\frac{d}{dt}V(t) = I(t) - \underbrace{(\eta\gamma R(t)V(t) + (1-\eta)\alpha V(t))}_{:=D(t)},$$

$$\frac{d}{dt}R(t) = \zeta\frac{V(t)^h}{\vartheta_{\mathrm{B}}^h + V(t)^h} - \beta R_i(t). \tag{2.1c}$$

In the differential equation model, the parameters $\alpha, \beta, \gamma, \zeta$ become rates and are restricted to be positive, although they may be greater than 1. The model choice parameter, however, remains restricted to $0 \leq \eta \leq 1$. (See Figure 1a for a graphical representation of the model.)

In the case of $\eta = 0$ (i.e., no spikiness) the full model, equation 2.1c, reduces to:

$$\dot{V} = I - \alpha V. \tag{2.1d}$$

2.2 Quantifying "Spikes". In the discrete time version of the GNM, "spikes" are determined by counting how often the membrane potential $V(t)$ (or decay $D(t)$) crosses the readout threshold ϑ_{R} from below. In multilabel classification, this value indicates class membership. In the case of a single pattern, we require it to cross the threshold exactly once. The error is then simply the difference between a target number of output spikes and the actual number of spikes during M time bins of a pattern.

In the continuous time case, we need to use a different way to quantify spiking based on the integral of the GNM membrane potential when it exceeds ϑ_{R}:

$$\mathcal{S} := \int_0^T \Theta(V(t) - \vartheta_{\mathrm{R}})V(t)\,dt, \tag{2.2}$$

where Θ denotes the Heaviside function, ϑ_{R} is a readout threshold, and T is the length of the trial. The error is then defined as the difference between the actual and the desired spike output (see Figure 2).

2.3 Training Algorithms. We use three different training algorithms to compare the GNM with the MST. First, we use a version of the eligibility-based ALL algorithm proposed by Gütig (2016), adapted to the GNM. We

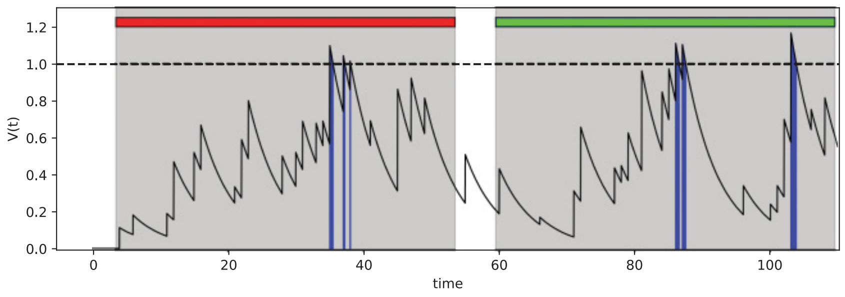


Figure 2: Example of pattern representations learned in a continuous-time GNM trained in an aggregate label setup. The integral of superthreshold membrane potential $\mathcal{S}$ is marked in blue. In this case, the continuous spike integral totals at ∼0.81 for the red pattern and ∼1.98 for the green pattern. In order to determine the spike indicator, we integrate the membrane potential when it is above a threshold. The total integral then indicates the number of spikes. In the case of the red pattern, the membrane potential crosses the threshold three times, but the total integral is ≈1; hence it indicates a single spike (rather than three spikes).

also show that there are alternative algorithms that can be applied to the GNM and provide better performance.

2.3.1 Aggregate-Label Learning (ALL). We first describe the eligibility-based learning algorithm similar to the one that Gütig (2016) proposed. We update the weights after a trial during which the neuron is shown target patterns and noise for a given period of time. Depending on what was shown, a target spiking number is then determined. After the trial is finished, we then set the error to a negative value if the neuron spiked too many times during this trial, to a positive value if it did not spike enough, and otherwise we do not update. Note that the algorithm does not provide any feedback about the degree to which the output was wrong, only about the "sign" of the error. The learning step proceeds by updating the weights in this way:

$$\Delta\omega_i = \begin{cases} \pm\lambda & \text{if } \varepsilon_i > D_9 \\ 0 & \text{if } \varepsilon_i \le D_9 \end{cases},$$

$$\varepsilon_i := \int_0^T I_i(t)V(t)\,dt. \tag{2.3}$$

Here, λ denotes the learning rate, which is positive when the error is positive and negative otherwise, and D_9 represents the ninth decile (top 10%) of the most eligible synapses. The variable ε_i is the eligibility of a presynaptic neuron i toward the postsynaptic neuron. It quantifies the extent to which the presynaptic neuron *i* has contributed to a spike of the postsynaptic neuron.

2.3.2 Error Trace Learning (ET). Second, we introduce an additional new training approach that uses precise information about the timing of erroneous spikes, as well as when and which feature patterns should have been recognized. The weight updates are calculated the same as in the ALL algorithm, except that now we base the eligibility and the error on the values of the integral at the time when each of the patterns was presented rather than after an entire episode of patterns and noise sequences. Thus, the algorithm obtains more detailed information about which weights caused erroneous spiking. This means that in response to a pattern with a target of two spikes, the neuron is supposed to cross the readout threshold exactly twice during the duration of the presented pattern. We calculate the error for each individual synapse by correlating its inputs with the error trace:

$$\begin{aligned}\Delta\omega_i &= \lambda\varepsilon_i,\\ \varepsilon_i &:= \int_0^T I_i(t)E(t)\,dt,\end{aligned} \tag{2.4}$$

where λ is again the learning rate and $E(t)$ denotes the error trace. Here, the variable ε_i should be understood as "error blame" of a presynaptic neuron *i* toward the postsynaptic neuron. It quantifies the extent to which the presynaptic neuron *i* has contributed to the erroneous activity of the postsynaptic neuron.

2.3.3 Error Trace Backpropagation. Finally, unlike traditional spiking models, where backpropagation can only be applied indirectly (Neftci, Mostafa, & Zenke, 2019; Tavanaei & Maida, 2017), the activation function of the GNM is differentiable and backpropagation can in principle be applied with no constraints. The temporal precision of error signaling in ET enables us to further extend it to training multilayered networks of GNMs. By way of demonstrating this, we show the network's ability to solve the multilabel classification task using an architecture of layered GNMs consisting of 10 hidden neurons (see Figure 3 for the architecture details).

2.4 Momentum Heuristic. In order to improve the speed of learning, we also use a momentum heuristic. During each learning step, we add a fraction of previous synaptic change to the update value:

$$\Delta\omega_i^{\text{current}} = \omega_i + \gamma\,\Delta\omega_i^{\text{previous}} \tag{2.5}$$

where γ is the momentum parameter; in all experiments it is set to 0.2.

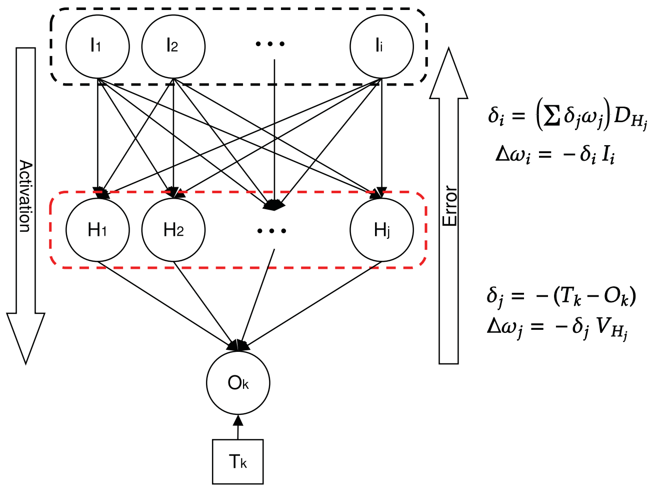

I1
I2
Ii
H1
H2
Hj
Ok
Tk
Activation
Error
$\delta_i = \left(\sum \delta_j \omega_j\right) D_{H_j}$
$\Delta\omega_i = -\delta_i I_i$
$\delta_j = -(T_k - O_k)$
$\Delta\omega_j = -\delta_j V_{H_j}$

Figure 3: Error backpropagation in multilayered network of the GNM. We introduce an intermediate layer of 10 hidden neurons. The hidden neurons have the same neural dynamics as the output neuron and connect to the input source in all-to-all fashion. It is worth noting that the output of the hidden neurons is presented to the output neuron as a sum of decaying currents in the continuous form. In order to successfully solve this task, we need to be able to propagate the error back through the network and correctly adjust the weights for both the output neuron and the hidden layer. The hidden layer neurons in the red dashed box are subject to continuous lateral inhibition.

3 Results

3.1 Aggregate Label Learning in the GNM. We first tested how well the discrete GNM can learn a single spatiotemporal pattern. Such a pattern is a temporal sequence of M binary strings of length N. Throughout this letter, we have kept $N = 100$ and $M = 50$. Patterns were generated randomly by drawing each of the bits from a Bernoulli distribution with $p(1) = 0.005$. In addition to randomly generated but fixed patterns, we expose the neuron to a stream of noisy background activity. The random activity is generated in the same way as the patterns, but unlike it, the noise is produced at each time bin. As a consequence, the statical properties of noise and pattern are identical in this setup.

The first task we set is as follows: GNM should respond with exactly one spike if the input is a pattern and should stay inactive otherwise (i.e., if presented with noise). Unlike the MST, the GNM does not have discrete output spikes. In order to interpret the output of the GNM, we thus need to set an (arbitrary) readout threshold value ϑ_R. The response of the GNM is determined by the number of times membrane potential $V(t)$ (or decay $D(t)$) crosses ϑ_R from below within the duration of the pattern (see Figure 1b). This number is used to indicate class membership. In the case of a single pattern, we require it to cross the threshold exactly once.

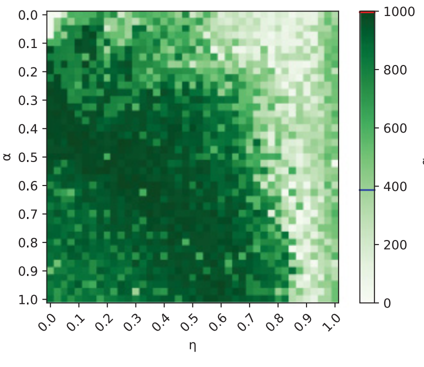
We train the neuron using the ALL algorithm (see section 2.3.1). We first test the performance of the GNM on the task of learning a single pattern. Here, the neuron is presented with a random number of spatiotemporal patterns embedded into noisy background activity at random times. At the end of a trial, the GNM receives feedback indicating whether it has released too many or too few spikes. In all experiments, we use the following parameters: $\beta = 0.3$, $\zeta = 1$, $\gamma = 1$, which allow the neuron to exhibit a postspike reset closely resembling that of the MST. For each target pattern, we sampled 41 different values of both α and η parameters (altogether 1681 parameter combinations) in order to test the performance of the GNM. We varied systematically the model choice parameter η from 0 (no spikiness) to 1 (complete spikiness) and the decay rate α from 0 (complete memory) to 1 (no memory). For each combination of parameters, we trained the GNM over 60,000 epochs (trials consisting of a random number of patterns embedded into randomly generated noisy background activity) with a learning rate $\lambda = 0.0001$.

In order to determine the quality of learning, we subjected the trained GNM to a stream of noise with randomly interspersed target patterns. If the GMM is working correctly, it should respond not to the noise but to the pattern. In practice, GNMs will not function perfectly. In order to quantify the classification reliability of the neuron, we recorded the number of random inputs given to the GNM before the GNM failed and averaged this number over 100 repetitions. We henceforth refer to this as the *noisy performance measure* and use it as an indicator of the quality of the GNM solution. Here, a higher noisy performance is better.

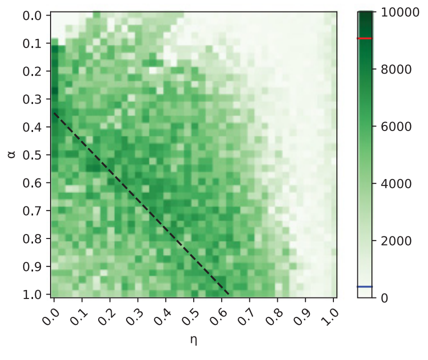
Figure 4 summarizes the noisy performance of the GNM averaged over five different patterns representing altogether 8405 different training iterations of the GNM. The graphical representation reveals a qualitative structure of the parameter space, which we find to be generic for any number of patterns.

For $\alpha, \eta \approx 0$, corresponding to the top left corner, noisy performance is low (i.e., the GNM does not classify well). The reason for the poor classification can be understood easily. In this region, the decay of the membrane potential $V(t)$ is low and the GNM integrates over all past events. The membrane potential remains in a permanent superthreshold state and thus is unable to cross the threshold ϑ_R from below (or indeed from above).

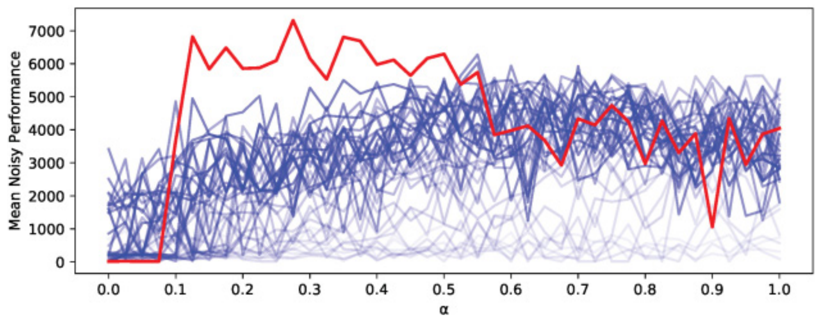
Allowing some leak by increasing α while keeping η at 0 (i.e., going down the left-most column in Figure 4) improves the performance dramatically. For example, α is adjusted from 0.05 to just 0.08, the noisy performance increases from approximately 0 to the global best. As α approaches 1, the performance decreases again. Therefore, for the subfamily of GNM



(a)



(b)



(c)

Figure 4: Mean noisy performance averaged over five training iterations as a function of α and η. The value reported is the average number of epochs a neuron can withstand without making an error, capped at (a) 1000, and (b) 10,000 time bins. Each simulation was trained for 60,000 epochs. The corresponding MST performance averaged at ~377 epochs and is marked on the color bar in blue. The red mark indicates the maximum value in the heat map. (b) The dashed line shows the estimated *optimal line* (see main text for explanation). (c) Noisy performance as a function of α. The red line indicates $\eta = 0$, and the blue lines show 40 other η settings.

models with $\eta = 0$, there must be a value of α that optimizes the learning, although this optimum is not well resolved. Note that in region $\eta = 0$, which we have considered so far, the neural dynamics is reduced to $V_i(t+1) = V(t) + I - \alpha V_i(t)$. As such, it lacks entirely the features that are usually associated with spiking neurons, including discrete spikes and an activation threshold.

A behavioral threshold ϑ_B is introduced to the GNM by increasing the spikiness parameter η. Figure 4 reveals that high performance of the model concentrates along a fuzzy line of combinations of α and η; we henceforth refer to this as the *optimal line* (see Figure 4b). However, performance along this line does not significantly increase for $\eta > 0$ relative to the nonspiking case of $\eta = 0$. Indeed, we can see that the GNM performance drops for extreme values of $\eta \approx 1$. In this regime, the threshold dynamics dominates, and the membrane potential is constrained to a small range of values, making learning impossible. Based on this, we conclude that at least for the case of a single pattern, introducing spikiness does not bring any benefits. Globally best performance can be achieved for $\eta = 0$ at $\alpha \simeq 0.3$.

There is also an appealing conjecture for the origins of the optimal line. We observe from equation 2.1a that the decay is effectively reduced by $(1 - \eta)$. The optimal line can then be interpreted as a consequence of there being an optimal value for the parameter α. To see this, assume that this optimal value is given by $\alpha = \alpha^*$. Assume further that the actual value of α is set to $\alpha' > \alpha^*$. A suitable choice of η satisfying $(1 - \eta) = \alpha^*/\alpha'$ can effectively offset the nonoptimal choice of α back to the optimal value. If that is true, it would generate precisely the observed optimal line in the parameter space portrait. Beyond this correction of the decay parameter, an increased spikiness has no apparent benefit.

3.2 Multipattern Learning in the GNM. So far, we have tested only how the GNM learns a single pattern. The key achievement of the MST is that a single neuron can learn to recognize multiple patterns and multiple classes of patterns. For example, there may be a set of patterns to which the MST responds with one spike and a set of patterns to which it responds with two spikes and so on. We now test whether the GNM can do the same. Similar to before, we interpret the GNM as "spiking" n times if during the presentation of the pattern, the membrane potential crosses the readout threshold ϑ_{R} from below n times.

Using this convention, we found that the multipattern case shows a qualitatively similar picture in parameter space as the single pattern case (see Figure 5), including an optimal line. Altogether, however, the noisy performance of the GNM dropped quickly with the number of pattern classes. For example, in the case of four different classes, the GNM responds to noise, and thus fails, after approximately 200 time bins on average, in the best case (see Figure 5c). Again, as in the single class case, there does not appear to be any benefit in increasing η above 0. While for some patterns the performance of the GNM with $\eta > 0$ is better, there was no consistent best value of η, and the best model with $\eta = 0$ was always comparable to the globally best result (see Figures 5b and 6). In order to understand this in more detail, we plot the noisy performance residuals graph (see Figure 6b and supplementary information S2). Here, we define residuals as a performance difference of the best α for any given η with the best performer from $\eta = 0$. In simple terms, for each column in a heat map, we select the best row and compare it to $\eta = 0$. Thus, negative values of the y-axis (i.e., points

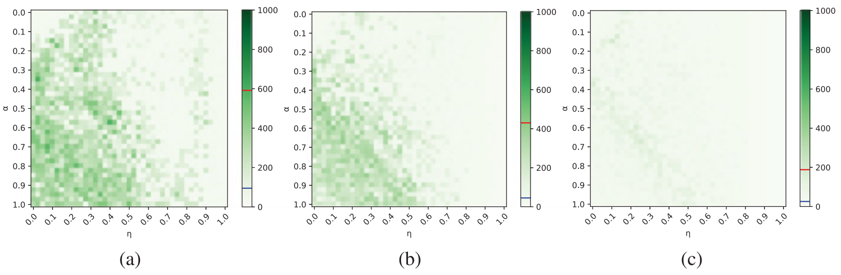


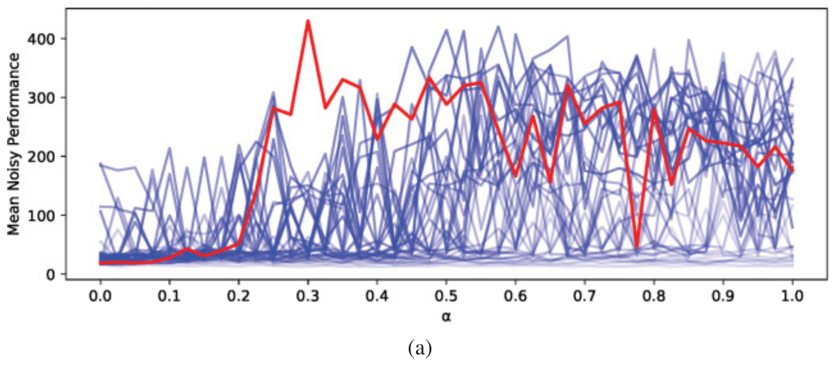
Figure 5: Same as Figure 4, but for (a) two, (b) three, and (c) four classes of patterns.

below the dashed line) indicate that the best performer for a particular η was worse than that of $\eta = 0$, for a given set of feature patterns. We found that most of the points fall under the dashed line, indicating that $\eta = 0$ is competitive. One of the training simulations has exceeded the line. Given that the setup is identical for all simulations, apart from the random seed and the pattern, we take this as an indication that there is no fundamental performance difference between the parameters with $\eta = 0$ and those with a positive η. In summary, we found that the GNM can perform well on the multilabel classification task with performance comparable to the more complicated MST model (indicated by the blue line on the color bars in Figures 4 and 5). Intriguingly, we also found that it is sufficient to consider the simplest nonspiking case of the GNM corresponding to $\eta = 0$.

3.3 Comparison to Other Training Methods and Neural Models. We have found that the GNM is competitive with the MST. However, the comparison is unfair because we compared a large number of simulations to just a single parameter setting of the MST. We now investigate the performance of the GNM relative to other models with more rigor.

To do this, we conducted 50 training simulations for the task of recognizing two patterns and discovered that MST outperformed the GNM (trained using ALL) only three times, for fixed parameters of the GNM—$\eta = 0.0$ and $\alpha = 0.3$—and MST: $\tau_m = 20$, and $\tau_s = 5$ (see Figure 7). This allows us to suggest that the GNM is not only simpler to implement but also learns better.

Next, we performed the same comparison with the LIF neuron. The LIF neuron is similar to the GNM, and it is reasonable to conjecture that it can be trained to perform multilabel classification as well. The MST and the GNM differ from the LIF in two features: they do not have a refractory period, and their reset function following a spike is exponential, rather than an absolute reset to the resting potential. We compared the ability of the LIF neuron to recognize patterns with the GNM assuming $\eta = 0$ and $\alpha = 0.3$ (see Figure 8). We used the same parameters for the LIF neuron and varied the length



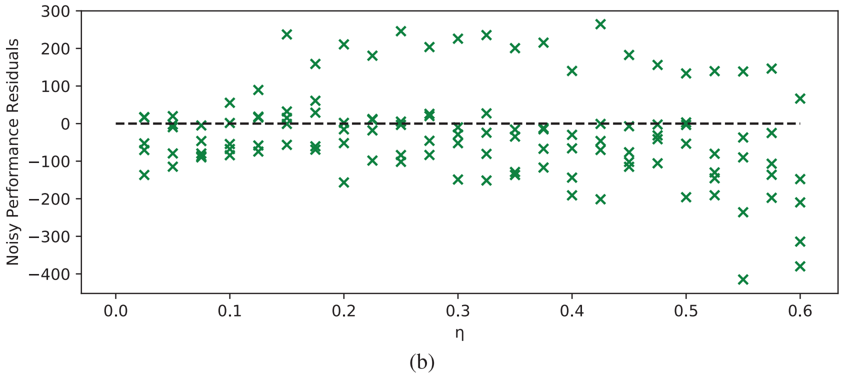


Figure 6: (a) Noisy performance as a function of α for same data as in Figure 5. The red line indicates $\eta = 0$, and the blue lines show 40 other η settings. (b) Noisy performance residuals for each of five training simulations as a function of η in a range from 0.025 to 0.6. Residuals as a performance difference of the best α for any given η with the best performer from $\eta = 0$. The deviation from the dashed line indicates the difference in performance in comparison to $\eta = 0$. Negative values indicate that the best training for a particular η was worse than $\eta = 0$ (see supplementary information S2 for the same graph for one, two, and four classes of patterns). We find that only in a single run was $\eta = 0$ suboptimal, thus conclude that there is a variation of performance that depends on the random seed given for pattern generation.

of the refractory period from 0 to 25. We contrast the two neural models by calculating the noisy performance ratio, which is the average noisy performance of the LIF neuron divided by that of the GNM in the same task. For a single pattern, the LIF neuron performs comparably to or slightly worse than the MST and the GNM, and the performance tends to increase with the length of refractory period (see Figure 8). However, for more than one pattern, the LIF neuron normally yielded a worse noisy performance than

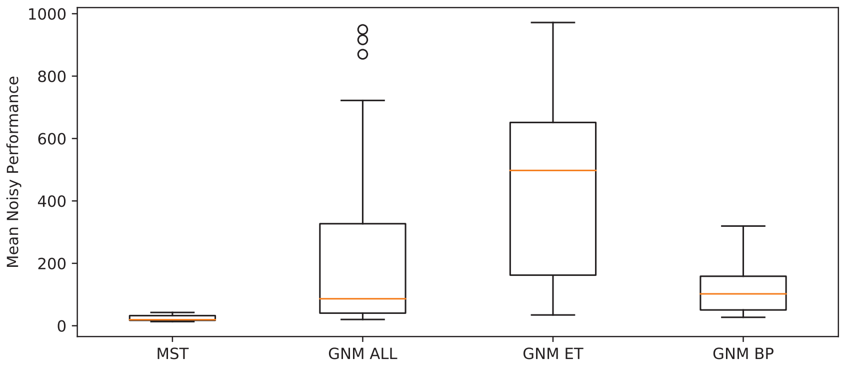


Figure 7: Comparison of the noisy performance measure in the task involving classification of two classes of patterns for MST, GNM trained using ALL, GNM trained using ET, and multilayered network of GNMs trained using BP. Neuron models have been trained for 60,000 epochs with the following parameters: GNM: $\alpha = 0.3$, $\eta = 0$, $\beta = 0.3$; MST: $\tau_m = 20$, and $\tau_s = 5$.

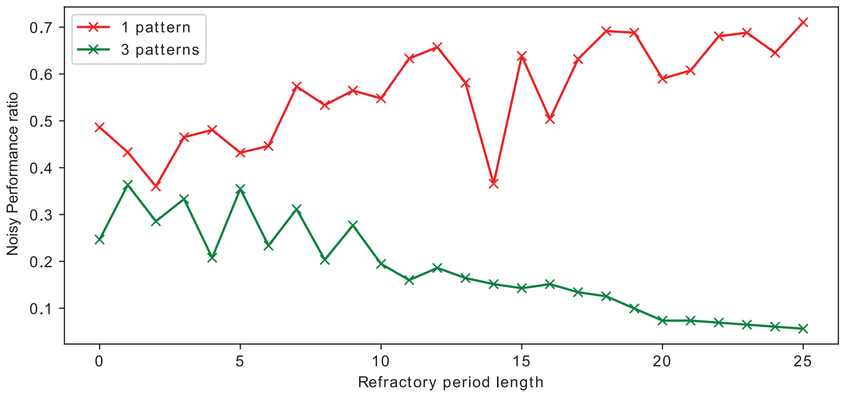


Figure 8: Noisy performance ratio of the GNM with $\eta = 0$ with a corresponding LIF neuron on the task of learning one and three classes of patterns. The noisy performance ratio is defined as the average noisy performance of the LIF neuron divided by that of the GNM (ALL). Both neuron models have been trained for 60,000 epochs with the following parameters: GNM: $\alpha = 0.3$, $\eta = 0$, $\beta = 0.3$; LIF: $\alpha = 0.3$, and varied length of the refractory period.

both MST and GNM. In this case, the performance drops with the increase in refractoriness. This shows that the period of forced inactivity hinders the multi-spike response. Note that for a refractory period of 0 the LIF neuron is identical to the GNM with the exception of the hard reset following a spike, but still performs worse on multilabel classifications. This suggests that the hard reset hinders multilabel classification and is the reason for reduced performance of the LIF model. This is consistent with our conjecture that the temporal autocorrelation of membrane potential is important for learning. We see from Figure 8 that the noisy performance increases with the refractory period. This is a consequence of the fact that during that period, the LIF neuron remains insensitive to inputs.

In addition to comparing the GNM to other neural models trained using the same algorithm, we also contrasted the performance of the original ALL algorithm with training techniques involving more information about the time of the error (i.e., error trace learning; see section 2.3.2). We find that the error trace learning algorithm has consistently outperformed ALL (see Figure 7). Out of 50 training simulations that we conducted (for a GNM neuron with parameters $\eta = 0, \alpha = 0.3$), 46 achieved a better result in terms of noisy performance when using the ET method. This tells us that despite the fact that the ALL is an elegant and simple training rule, it is also suboptimal.

Moreover, we propose an extension of the ET rule to setups of multilayer networks. The error backpropagation algorithm (see section 2.3.3) can be applied directly to the GNM (see Figure 7). However, for the present task, we could not find any benefits in applying backpropagation. This is not to say that for more complex problems, backpropagation may be beneficial in SNNs. Exploring this is beyond the scope of this letter, and we leave it to future research.

3.4 Interpreting GNM as a System of Chemical Reactions. A common assumption in the SNN literature, including the MST, is that the input channels are clocked that is, the model is updated in discrete time. It is straightforward to extend the GNM model to the continuous case (corresponding to equation 2.1c). We found that training the model in continuous time yielded qualitatively the same results as the discrete time case. We demonstrate the feasibility of this interpretation by training a continuous-time version of the GNM to recognize two classes of patterns (see Figure 2). The extension to the continuous case is interesting because then the GNM model with $\eta = 0$ (see equation 2.1d) can be interpreted as the description of a molecular species V that decays with a rate of α. The input I is then mathematically equivalent to N different input chemical species I_i, each decaying to V with a rate of $w_i C$ and to a null species with a rate of $(1 - w_i)C$. In this case, the constant C sets the time scale of the decay and could be the same for all presynaptic neurons. Having interpreted the GNM as a chemical system, we can then test its ability to recognize patterns by solving the differential equation 2.1d.

An underlying assumption of the differential equation models is that the number of molecules involved in the system is very large (technically infinite), such that V can be described as a concentration. In any real system, the number of particles is finite. Indeed, in many biological information processing tasks, there may only be a small number of particles involved in the

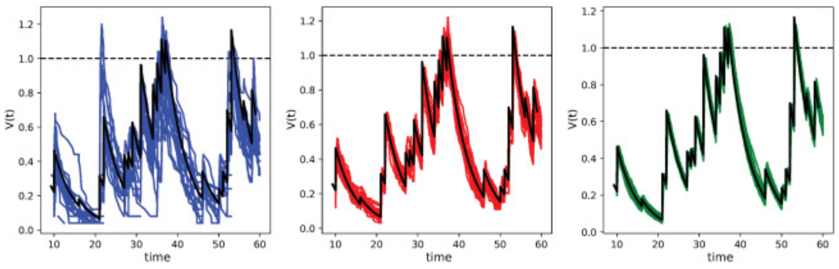


Figure 9: Membrane potential of a deterministic continuous-time GNM neuron trained to recognize two classes of patterns (black line) and equivalent stochastic chemical reaction network simulations with an input "spike" equivalent to 25, 100, and 500 molecules. For parameters $\eta = 0$, $\alpha = 0.2$, $\beta = 0.3$. The presynaptic spikes were encoded as an instantaneous increase of corresponding presynaptic "species" by N, where t^i_j is the time of the jth spiking event of the ith presynaptic neuron, and $N = 25, 100, 500$ is the number of particles that is added to the presynaptic species i at time t^j_i. C was set to 10.

computation. In this case, the system will exhibit noise around the exact solution of equation 2.1d (see Figure 9 for a comparison of the stochastic and the deterministic solution). A concrete consequence of this is that the output of the neuron becomes stochastic with more or less frequent incorrect outputs, depending on the number of particles.

4 Discussion

In this letter, we have probed the minimal ingredients necessary for neural computation in the context of multilabel classification of spatiotemporal patterns. We introduced the GNM, which can solve multilabel classification tasks at least as well as the MST, while being purely state based. The model also has a conservation of membrane potential built in. This does not preclude leakage of membrane potential, but it prevents its creation out of nothing. In that sense, the model is physically plausible, which allows it to be interpreted in terms of concrete implementations (see below).

The dynamics of the GNM are simple and its parameters easily interpretable, which supports an intuitive understanding of what precisely it is that makes single neuron classification work. The parameter α can be interpreted as a memory. It determines how much membrane potential is leaked between two update steps. In the extreme case of $\alpha = 1$, the neuron is reset during each time step and has no memory of past inputs. In this limit, the GNM is reduced to a standard rate-coding neuron. The performance of the GNM is substantially decreased, but some learning is still possible. For $\alpha = 0$, the neuron integrates over all past events and never forgets. It is clear from both basic considerations and our simulation results that this latter limit does not allow the GNM to recognize patterns. In between those two extremes is an optimal value for the memory of the GNM. Figures 4c and 6a suggest that the model is not particularly sensitive to the memory parameter, at least not for intermediate values. Interestingly, however, our simulations also suggest that at the lower end of the parameter range, there is a critical value of α that separates almost perfect ability to learn from complete nonperformance.

The second key parameter is the model choice parameter η. It controls the extent to which the neural dynamics is affected by an internal threshold and hysteresis—in short, how much spikiness the neural dynamics exhibits. For $\eta = 0$, the internal dynamics is a simple exponential decay with time constant α. No thresholds are defined internally, and there is no spiking whatsoever. Note, however, that the use of the neuron still requires an evaluation threshold ϑ_{R} to be set in order to be able to interpret the membrane potential of the GNM as indicating the pattern class.

Increasing η introduces an additional behavioral threshold parameter, ϑ_{B}, which now *does* have an impact on the internal dynamics of the GNM. As the membrane potential nears ϑ_{B}, an additional decay term R becomes relevant, such that after crossing the threshold, the decay may be higher than before crossing the threshold. This endows the GNM a spikiness and, most of all, with a time-limited memory of past spiking events

We found that model performance was consistently best along an off-center diagonal in the lower left quadrant of the heat maps (see Figures 4 and 5). Crucially, however, there is no consistent evidence for an optimal point along this diagonal. The conclusion to draw from this is that it is sufficient to consider the reduced parameter space corresponding to $\eta = 0$—the case of no spikes. Put differently, there does not appear to be any benefit in spiking.

The existence of the optimal line provides some insights into the necessary ingredients for spiking networks in that it points to the memory parameter α as the main determinant of performance. The optimal line, albeit not very well defined in our models, is the line of constant memory, because the factor $(1 - \eta)$ effectively reduces the memory parameter α. Therefore, it appears from our simulations that there is an optimal memory for the performance of the GNM which lies in between the extreme and nonperforming cases of $\alpha = 0$ and $\alpha = 1$ corresponding to no forgetting and no memory at all. However, note that the model is not particularly sensitive to the precise value of α, such that there is a range of values for which performance is good.

The conclusion that the GNM can perform with no spiking opens an interesting perspective. For $\eta = 0$, the GNM model, equation 2.1d, looks formally like the time evolution of the concentration V of a molecular species subject to decay, $V \xrightarrow{\alpha} \varnothing$, plus occasional instantaneous increases of V,

$V \longrightarrow kV$. Fundamentally, such a chemical system is an extremely simple system that can be implemented easily in (wet) experiments. Yet as we show, this simple system is sufficiently rich in its dynamics in order to perform the multilabel classification as well as the specialized MST model. All the chemical system retains in common with the MST is the temporal correlation of the input. This leads us to conjecture that this temporal autocorrelation is a crucial element for multilabel classification of spatiotemporal patterns.

This formal equivalence of GNM and chemical systems begs the question whether there actually are man-made or naturally occurring chemical systems that recognise spatiotemporal patterns. An obvious place to look for such systems are biochemical networks. It is conceivable that multilabel classification is exploited by gene regulatory networks to control gene expression by means of sequences of gene expression events.

Having shown that very simple systems can perform multilabel classification, it is instructive to compare the GNM and MST to another very simple model of a spiking neuron: the LIF neuron. The LIF neuron is different from both the GNM and the MST in that it has a hard reset following a spike and typically undergoes a refractory period. The refractory period, together with lateral inhibition, is a useful feature in the context of STDP learning (Feldman, 2012; Gerstner & Kistler, 2002b), which can be used to facilitate a winner-takes-all dynamics in multilayer SNN networks, which in turn is important to prevent all postsynaptic neurons from learning the same parts of the input. Beyond that, it is unclear whether there is a computational benefit in the refractory period.

Our simulations showed (see Figure 8) that the LIF has a comparable performance to the MST/GNM when classifying a single pattern, but its ability to learn drops for multipattern classification. This is understandable because the refractory period effectively shortens the time the LIF is able to react to incoming signals, thus making it hard for the neuron to activate several times during a limited period. Yet as our simulations show (see Figure 8), the performance of the LIF neuron is worse even for a refractory period of length 0. Once the refractory period is removed, the only remaining difference between the LIF and the GNM is the hard reset. Note that this hard reset effectively destroys the temporal autocorrelation of the membrane potential. Hence, the observation that the LIF neuron performs worse than the GNM supports further our conclusion that a balanced memory of the membrane potential is required for good performance on multilabel classification of spatiotemporal signals. This now raises the question whether of biological neurons, which clearly do have a refractory period, are suboptimal components. We do not believe that this conclusion can be made because brains operate in a different context from the restricted problem set that we considered here. Moreover, the refractory period in real neurons may well be a reflection of some resource limitations or physical constraints that we have not considered here, thus making a comparison invalid.

Throughout this letter we have evaluated the GNM assuming an aggregate label delayed feedback learning rule during training. This training method is mainly motivated by its biological plausibility. In applications of the GNM/MST in the context of AI, biological plausibility is not a relevant criterion. We found, perhaps rather unsurprising, that dropping the requirement of aggregate label delayed feedback in favor of more immediate and information-rich feedback led to increased model performance (see section 3.3).

Once we allow such direct error feedback, we can extend it to backpropagation-based training methods in networks of GNMs. We demonstrate the feasibility of this approach by solving the multilabel classification task using a layered network of GNMs with 10 hidden neurons (see Figures 3 and 7). Deep learning with SNNs could lead to substantive benefits in terms of smaller models and more efficient hardware if only it is possible to transfer established deep learning techniques to spiking architectures. We leave it to future research to establish whether the GNM or similar spiking architectures could indeed be a credible alternative for existing deep architectures.

5 Conclusion and Outlook

Gütig's multispike tempotron is a powerful single neuron model that can classify spatiotemporal patterns into multiple classes. The model is also complicated to implement. Here, we showed that the much simpler GNM neuronal model can achieve the same performance as the MST. Our results indicate that the important feature of neuronal models is the temporal autocorrelation of the membrane potential; that is, how quickly the neuron forgets about past inputs. We found that for intermediate values, the model performance is maximized. It remains an open question for future research whether this conclusion is specific to the particular task we considered, or whether the optimal memory emerges as *the* crucial parameter in all applications. If the power of SNNs is to be leveraged in practical AI applications, then it will be necessary to understand the minimal spiking neuron that is sufficient for a particular task so as to be able to build resource-efficient systems.

References

Brunel, N., & van Rossum, M. C. W. (2007). Lapicque's 1907 paper: From frogs to integrate-and-fire. *Biological Cybernetics*, *97*(5), 337–339.

Feldman, D. E. (2012). The spike-timing dependence of plasticity. *Neuron*, *75*(4), 556–571.

Florian, R. V. (2012). The chronotron: A neuron that learns to fire temporally precise spike patterns. *PLOS One*, *7*(8).

Gerstner, W., & Kistler, W. M. (2002a). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge: Cambridge University Press.

Gerstner, W., & Kistler, W. M. (2002b). Mathematical formulations of Hebbian learning. *Biological Cybernetics*, *87*(5–6), 404–415.

Gerstner, W., Kistler, W. M., Naud, R., & Paninski, L. (2014). *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge: Cambridge University Press.

Gütig, R. (2016). Spiking neurons can discover predictive features by aggregate-label learning. *Science*, *351*(6277), aab4113.

Gütig, R., & Sompolinsky, H. (2006). The tempotron: A neuron that learns spike timing–based decisions. *Nature Neuroscience*, *9*(3), 420–428.

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *Journal of Physiology*, *117*(4), 500–544.

Indiveri, G., Linares-Barranco, B., Hamilton, T., van Schaik, A., Etienne-Cummings, R., Delbruck, T., . . . Boahen, K. (2011). Neuromorphic silicon neuron circuits. *Frontiers in Neuroscience*, *5*, 73.

Izhikevich, E. M. (2003). Simple model of spiking neurons. *IEEE Transactions on Neural Networks*, *14*(6), 1569–1572.

Jolivet, R., Lewis, T. J., & Gerstner, W. (2003). The spike response model: A framework to predict neuronal spike trains. In O. Kaynak, E. Alpaydin, & E. Oja (Eds.), *Lecture Notes in Computer Science: Vol. 2714. Proc. Joint International Conference ICANN/ICONIP 2003* (pp. 846–853). Berlin: Springer.

Lin, C., Wild, A., Chinya, G. N., Lin, T., Davies, M., & Wang, H. (2018). Mapping spiking neural networks onto a manycore neuromorphic architecture. *SIGPLAN Notices*, *53*(4), 78–89.

Maass, W. (1997). Networks of spiking neurons: The third generation of neural network models. *Neural Networks*, *10*(9), 1659–1671.

Mahajan, D., Girshick, R. B., Ramanathan, V., He, K., Paluri, M., Li, Y., . . . van der Maaten, L. (2018). *Exploring the limits of weakly supervised pretraining*. CoRR, abs/1805.00932.

Mohemmed, A., Schliebs, S., Matsuda, S., & Kasabov, N. (2012). Span: Spike pattern association neuron for learning spatio-temporal spike patterns. *International Journal of Neural Systems*, *22*(4), 1250012.

Neftci, E. O., Mostafa, H., & Zenke, F. (2019). *Surrogate gradient learning in spiking neural networks*. CoRR, abs/1901.09948.

Plana, L. A., Clark, D., Davidson, S., Furber, S., Garside, J., Painkras, E., . . . Bainbridge, J. (2011). Spinnaker: Design and implementation of a GALS multicore system-on-chip. *J. Emerg. Technol. Comput. Syst.*, *7*(4), 17–18.

Ponulak, F. (2005). *ReSuMe: New supervised learning method for spiking neural networks* (Technical Report). Poznan, Poland: Institute of Control and Information Engineering, Poznan University.

Radford, A., Wu, J., Child, R., Luan, R., Amodei, D., & Sutskever, I. (2018). Language models are unsupervised multitask learners. *Journal of Machine Learning Research*, *12*, 2493.

Rajendran, B., & Alibart, F. (2016). Neuromorphic computing based on emerging memory technologies. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, *6*(2), 198–211.

Rubin, R., Monasson, R., & Sompolinsky, H. (2010). Theory of spike timing–based neural classifiers. *Physical Review Letters*, *105*(21), 1–4.

Shahsavari, M., Devienne, P., & Boulet, P. (2019). Spiking neural computing in memristive neuromorphic platforms. In Leon Chua, Georgios Sirakoulis, & Andrew Adamnatsky (Eds.), *Handbook of memristor networks.* Berlin: Springer.

Tavanaei, A., & Maida, A. S. (2017). *BP-STDP: Approximating backpropagation using spike timing dependent plasticity*. CoRR, abs/1711.04214.

Wunderlich, T., Kungl, A. F., Müller, E., Hartel, A., Stradmann, Y., Aamir, S. A., . . . Petrovici, M. A. (2019). Demonstrating advantages of neuromorphic computation: A pilot study. *Frontiers in Neuroscience*, *13*, 260.

Yu, Q., Tang, H., Tan, K. C., & Li, H. (2013). Precise-spike-driven synaptic plasticity: Learning hetero-association of spatiotemporal spike patterns. *PLOS One*, *8*(11), 1–16.

Received October 20, 2019; accepted March 5, 2020.