Continuous-Time Spike-Based Reinforcement Learning for Working Memory Tasks

Marios Karamanis, Davide Zambrano, and Sander Bohté(✉)

CWI, Machine Learning Group, Amsterdam, The Netherlands
{marios,davide,sbohte}@cwi.nl

Abstract. As the brain purportedly employs on-policy reinforcement learning compatible with SARSA learning, and most interesting cognitive tasks require some form of memory while taking place in continuous-time, recent work has developed plausible reinforcement learning schemes that are compatible with these requirements. Lacking is a formulation of both computation and learning in terms of spiking neurons. Such a formulation creates both a closer mapping to biology, and also expresses such learning in terms of asynchronous and sparse neural computation. We present a spiking neural network with memory that learns cognitive tasks in continuous time. Learning is biologically plausibly implemented using the AuGMeNT framework, and we show how separate spiking forward and feedback networks suffice for learning the tasks just as fast the analog CT-AuGMeNT counterpart, while computing efficiently using very few spikes: 1–20 Hz on average.

Keywords: Reinforcement learning · Working memory
Spiking neurons

1 Introduction

Reinforcement Learning [17] describes how animals can learn to act effectively given sparse and possibly delayed rewards from their environment. For many tasks, optimal action selection requires some form of memory: the shortest path to a parked car relies on remembering where the car was parked, and understanding text requires the integration of information over the length of the sentence, if not from earlier paragraphs. For event-based and discrete-time optimization problems, Reinforcement Learning has been used to successfully train deep [11,16] and recurrent neural networks [1]. For working memory tasks, [1] demonstrated that LSTMs can be trained with the RL Advantage Learning algorithm, but this type of “off-policy” RL based on error-backpropagation is considered biologically implausible given the preponderance for “on-policy” RL like SARSA [12]. How animals can learn such tasks with SARSA-like RL and neural network models has been the topic of much research in neuroscience, with implications also in fields like deep learning and neuromorphics.

Recent work [15,20] has suggested how working memory tasks can be learned in neural network models equipped with memory neurons, where memory neurons learn which stimuli need to be remembered for later use; learning is then made local and plausible using feedback connections [13]. While standard RL is formulated in an event-based manner, that is, framed in terms of state-changes, animals operate in a continuous-time setting and Zambrano et al. showed in [20] that a continuous-time version of AuGMenT (CT-AuGMenT) can be realized using an action selection mechanism that integrates evidence - drawing inspiration from the brain's basal ganglia structures - combined with a separate feedback network for learning. Missing so far is a model of biologically plausible RL based on spiking neurons: here we present such a model, and we show how learning can in fact be based on the (sparse) relative timing of spikes.

We show how the CT-AuGMenT framework can be extended to asynchronous and sparsely active spiking neural networks. Recent work has shown how spiking neurons can be used to computed convolutional neural networks [5,18] and compute control [7]; RL versions are lacking. We turn to adaptive spiking neurons [2] and develop two spike-based approaches: the first where spikes carry approximations of both forward and feedback signals, and the CT-AuGMenT-derived learning mechanism uses these signal approximations. In the second, we develop spike-triggered learning by exploiting the fact that the dynamics of the tasks are much slower than the timescale of timesteps in the simulation, and CT-AuGMenT weights-updates can be approximated by sparse sampling of the learning components – spike-triggered learning then uses the asynchronous nature of adaptive spike-coding where changes in signals elicit more spikes in the network, and hence higher precision sampling.

We show how these approaches can be applied to two standard RL working memory tasks (T-Maze and Saccade-anti-Saccade), and find that networks trained with both spike-based learning methods successfully and efficiently learn the tasks. When using spike-based learning, we find that very low firing rates in the network suffice, where the spike-triggered learning approach requires only slightly higher firing rates, as can be expected since so very learning events take place. Together, we demonstrate spiking neural networks to learn cognitive tasks, capable of online-learning using sparse spike-triggered learning.

2 CT-AuGMEnT

In [14,15], AuGMEnT was developed as an artificial neural network (ANN) implementation of the on-policy SARSA reinforcement learning algorithm for solving Markov Decision Processes (MDPs) that require learnable working memory to construct Markov states in the hidden layer of the neural network model. AuGMEnT implements a biologically plausible local learning rule based on four factors: attentional feedback, forward activation, the local derivative of the transfer function, and a global neuromodulatory scalar value that signals the temporal difference error (TD-error) δ (Fig. 1a). This learning rule is local and enables the learning of XOR-like non-linear function mappings in multi-layer networks [15].

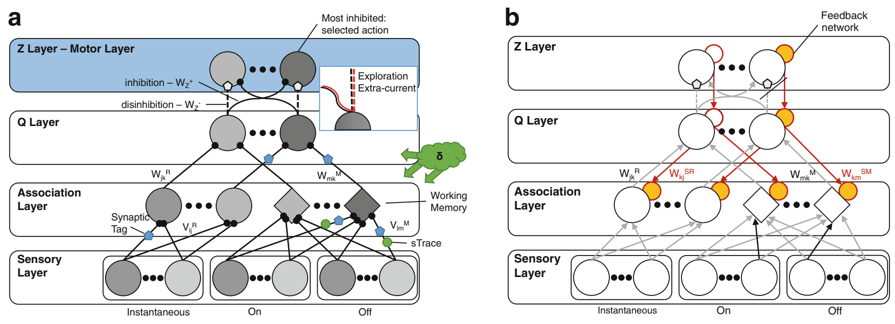


Fig. 1. (a) The CT-AuGMEnT architecture. The feedforward layers include memory units in the association layer (diamonds) to compute Q-values in the Q-layer. The Q-values are integrated in the action-selecting Z-layer, where the most inhibited action is selected at any point in time. Feedback from the (sole) selected action induces tags and traces on the synapses, which in combination with TD-error (δ) determines changes in synaptic weights. (b) In continuous-time, the feedback activity from the selected action is carried by a separate feedback network with its own weights (orange network). (Color figure online)

In [19,20] the CT-AuGMEnT framework was developed as an extension of AuGMEnT to include a realistic notion of continuous-time, introducing a dynamic action selection system and demonstrating an explicit feedback network with layer-wise delays and separately learned feedforward and feedback weights. The inclusion of an action selection system decouples the typical timescale of actions from the time resolution of the simulation, allowing for continuous-time on-policy SARSA learning. The resulting network is depicted in Fig. 1b.

As described in [20], the CT-AuGMENT network comprises of four layers (Fig. 1a, b): a sensory input layer, a hidden "association" layer, a Q-layer, and an action layer Z. In the sensory layer, instantaneous units directly represent the stimulus intensity $x(t)$, and transient "on/off"units represent positive and negative changes in stimulus intensity, $x^+(t)$ ("on") and $x^-(t)$ ("off"):

$$x^+(t) = \frac{1}{dt}[x(t) - x(t-dt)]_+, \qquad x^-(t) = \frac{1}{dt}[x(t-dt) - x(t)]_+, \tag{1}$$

where $[.]_+$ is a thresholding operation returning 0 for negating inputs. The hidden layer is comprised of regular units and memory units, where the instantaneous units i connect to the regular units j via connections v_{ij}^R and the transient units l connect to the memory units m via connections v_{lm}^M. Activations are then computed as:

$$a_j^R(t) = \sum\nolimits_i v_{ij}^R x_i(t) \quad y_j^R(t) = f(a_j^R(t)) \tag{2}$$

$$a_m^M(t) = a_m^M(t-dt) + \sum\nolimits_l v_{lm}^M x_l'(t) \quad y_m^M(t) = f(a_m^M(t)). \tag{3}$$

where $f(.)$ denotes the neuron's transfer function, here the standard sigmoid transfer function; for brevity of notation, $x_l'(t) = [x^+(t) \quad x^-(t)]$. The third layer is connected to the association layer via connections w^M_{mk} and w^R_{jk}, computing Q-values for every possible action k in the current state s, $q_k(t)$:

$$q_k(t) = \sum_m w^M_{mk} y^M_m(t) + \sum_j w^R_{jk} y^R_j(t). \tag{4}$$

The Z-layer, modeled after action selection in the basal ganglia [8], implements an action-selection model based on competition between possible actions by connecting the Z-layer to the Q-layer with off-center on-surround connectivity: each q-unit inhibits its corresponding Z-unit and excites all other Z-neurons (Fig. 1a, top). The input to a Z-layer unit u_i is thus:

$$u_i(t) = -w^- q_i(t) + w^+ \sum_{j \neq i}^{n} q_j(t), \tag{5}$$

where we set $w^-/w^+ = \nu$, with ν the number of possible actions in the task; the activation of the Z units can then be modeled as a leaky integrator:

$$\dot{a}_i(t) = -\rho(a_i(t) - u_i(t)), \tag{6}$$

where ρ is a rate constant that determines how fast equilibrium is reached. The Z-layer output $y_i(t)$ is bounded using the sigmoid activation function:

$$y_i(t) = \sigma(a_i(t)). \tag{7}$$

The Q-layer thus determines the degree of inhibition in the Z-layer, where, somewhat counterintuitive, the selected action is the one that receives the most inhibition. Exploration is implemented as the addition of an external current to the explorative action unit in Eq. (5) [20].

Learning: In the CT-AuGMenT network, network plasticity is modulated by two factors: a global neuromodulatory signal and an attentional feedback signal. At every time-step, the Z-unit corresponding to the winning action a creates synaptic tags (equivalent to eligibility traces) by sending feedback activity to earlier processing levels. Tags in the Q-layer decay and are updated as:

$$Tag_{jk}(t + dt) = -\frac{1}{\phi} Tag_{jk}(t) + dt[y_j(t) z_k(t)], \tag{8}$$

with $z_k = 1$ for the selected action and $z_k = 0$ for the other actions. The association units that provided strong input to the winning action a thus also receive the strongest feedback. Tags - mimicking eligibility trace - on connections between regular units and instantaneous units are equivalently computed as:

$$Tag_{ij}(t + dt) = -\frac{1}{\phi} Tag_{ij}(t) + dt[x_i(t) f'(a^R_j(t)) w^R_{kj}], \tag{9}$$

where $f'(\cdot)$ denotes the local derivative of the transfer function f, and the feedforward connections w^R_{jk} and the feedback connections w^R_{kj} may have different

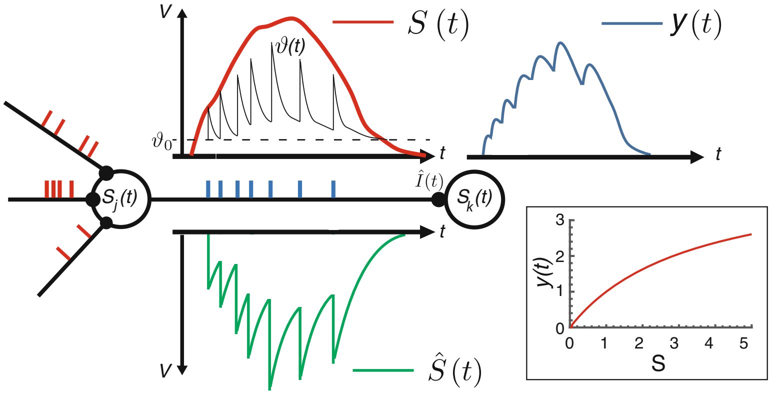


Fig. 2. ASN-based neural coding. Input spikes (red ticks), induce a smoothed activation $S(t)$ in the post-synaptic neurons. The neuron emits spikes (blue ticks) when the input activation exceeds a variable threshold $\vartheta(t)$, and a refractory response scaled by the momentary adaptation is subtracted from the activation at the time of spiking. The resulting total refractory response $\hat{S}(t)$ approximates the rectified activation $S(t)^+$. At the next target neuron, the emitted spike-train induces an (unweighted) activation $y(t)$; the transfer function (inset) describes the average relationship between the activation $S(t)$ and the target activation $y(t)$. (Color figure online)

strength [13]. Synaptic traces between sensory units l and memory cells m enable the proper learning of working memory:

$$\begin{aligned} sTrace_{lm}(t+dt) &= sTrace_{lm}(t) + dt[x'_l(t)] \\ Tag_{lm}(t+dt) &= -\tfrac{1}{\phi}Tag_{lm}(t) + dt[sTrace_{lm}(t)f'(a^M_m(t))w^M_{km}]. \end{aligned} \tag{10}$$

To implement on-policy SARSA temporal difference (TD) learning [17], the predicted outcome $q_a(T-1)$ is compared to the sum of the reward $r(t)$ and the discounted action-value $q_{a'}(T)$ of the unit a' that wins the competition at time T, resulting in a TD error $\delta(T) = r + \gamma q_{a'}(T) - q_a(T-1)$. For continuous-time TD learning, [20] gives the following TD error:

$$\delta(t) = r(t) + \frac{1}{dt}\left[\left(1 - \frac{dt}{\tau}\right) q_{a'}(t) - q_a(t-dt)\right], \tag{11}$$

with learning rate β, weight updates are then defined as:

$$\begin{aligned} v_{ij}(t+dt) &= v_{ij}(t) + dt[\beta\delta(t)Tag_{ij}(t)], \\ v_{lm}(t+dt) &= v_{lm}(t) + dt[\beta\delta(t)Tag_{lm}(t)], \\ w_{jk}(t+dt) &= w_{ja}(t) + dt[\beta\delta(t)Tag_{jk}(t)]. \end{aligned} \tag{12}$$

3 Adaptive Spiking Neurons

Adaptive Spiking Neurons (ASNs) [2] are a variant of standard Leaky-Integrate-and-Fire spiking neurons incorporating a fast multiplicative adaptation mechanism, where the fast adaptation limits the neuron's asymptotic firing rate. The ASN includes spike-triggered adaptation and a dynamical threshold that allows it to match neural responses while maintaining a high coding efficiency.

Illustrated in Fig. 2, adaptive spike-based neural coding is described as a Spike Response Model (SRM) [6], where the input to a neuron j is computed as a sum of spike-triggered post-synaptic currents (PSCs) from pre-synaptic input neurons i. The total PSC, $I(t)$, is computed as a sum over spike-triggered (normalized) kernels $\kappa(t_s^i - t)$ each weighted by synaptic efficacies w_{ij}:

$$I(t) = \sum_i \sum_{t_s^i} w_{ij}\, \kappa(t_s - t), \tag{13}$$

where t_s^i denotes the timing of spikes from input neuron i. A normalized exponential filter $\phi(t)$ is applied to $I(t)$ to obtain the neuron's activation $S(t)$:

$$S(t) = (\phi * I)(t). \tag{14}$$

In the SRM formulation [2], the membrane potential of the neuron is obtained as the neuron's activation $S(t)$ from which the total refractory response $\hat{S}(t)$ is subtracted, where $\hat{S}(t)$ is computed as the sum of spike-triggered refractory response kernels $\eta(t)$ each scaled by the (variable) value of the neuron's threshold at the time of spiking ($\vartheta(t_j)$); $\hat{S}(t)$ then approximates the rectified $S(t)$: $S(t)_+$.

A spike is emitted by neuron j at time t whenever $S - \hat{S}(t) > \theta(t)$ and the membrane potential is reset by subtracting a scaled refractory kernel $\eta(t)$ which is then added to the total refractory response $\hat{S}(t)$. Spike-triggered adaptation is incorporated into the model by multiplicatively increasing the variable threshold $\theta(t)$ with a decaying kernel $\gamma(t)$ at the time of spiking, and by controlling the speed of the firing rate adaptation using the multiplicative parameter m_f:

$$\theta(t) = \theta_0 + \sum_{t_s} m_f \theta(t_s)\gamma(t_s - t), \qquad \hat{S}(t) = \sum_{t_s} \theta(t_s)\eta(t_s - t). \tag{15}$$

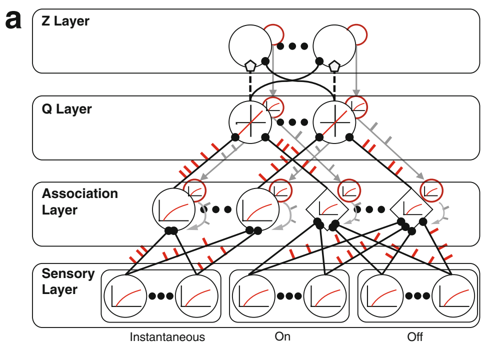
We set the PSC kernel as equal to the refractory response kernel $\eta(t)$, and model this kernel and the threshold kernel $\gamma(t)$ as decaying exponentials with corresponding time-constants τ_η, τ_γ; as is the membrane filter $\phi(t)$ (τ_ϕ):

$$\kappa(t) = \eta(t) = \exp\left(\frac{t_s - t}{\tau_\eta}\right), \tag{16}$$

$$\gamma(t) = \exp\left(\frac{t_s - t}{\tau_\gamma}\right), \qquad \phi(t) = \phi_0 \exp\left(\frac{t_s - t}{\tau_\phi}\right), \tag{17}$$

where the timing of outgoing spikes is denoted by t_s, θ_0 is the resting threshold.

Given a fixed input current $I(t)$ resulting in a fixed activation $S(t)$, the emitted spike-train from the post-synaptic neuron has an (unweighted) fixed size impact $y(t)$ on the next target neuron. We characterize the relationship between activation $S(t)$ and target impact $y(t)$ as the effective ASN transfer-function (inset); this function has a half-sigmoid like shape and can be either computed analytically for particular parameter choices (i.e. [18]) or approximated. For the analog spike-like network in Sect. 4, we approximate the shape of this transfer-function with the positive rectified *tanh*() function: *tanhP*().



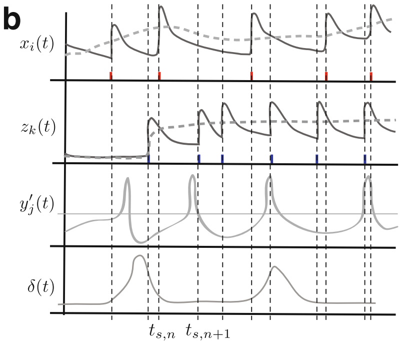


Fig. 3. (a) Spiking CT-AuGMent. Indicated by the half-sigmoid graphs are the neurons that are set to have *tanhP*() as transfer functions (in the analog rectified network), which are substituted by ASN neurons in the spiking network versions. Ticks along network connections indicate which part of the network "spikes". (b) Spike-based and spike-triggered learning: spike-based learning uses the analog global δ and local $y'(t)$ signals and those derived from feedforward spikes, $x(t)$ and feedback spikes, $z(t)$; spike-triggered learning considers those signals only at spike times $t_{s,n}$.

4 Spike-Based CT-AuGMenT

Analog Rectified CT-AuGMenT. To convert the CT-AuGMenT network to a spiking neural network, we replace the analog neurons by ASN models. The main obstacle here is that ASNs effectively have a rectified half-sigmoid-like transfer function, as illustrated in Fig. 2. The CT-AuGMenT network uses sigmoidal transfer-functions for the feedforward stage, and linear neurons for Q-layer and the feedback network [20]. While for instance [10,13] suggest that there is some flexibility with regard to the feedback network, we create an analog network where the neurons in the feedforward Sensory and Association layer use the *tanhP*() transfer-function, as well as the feedback network from the Q-layer projecting to the Association layer (illustrated in Fig. 3a). We train this network on the tasks to ascertain the feasibility of training spike-based networks with rectified half-sigmoid-like transfer functions.

Spike-Based Learning. Spiking-AuGMenT incorporates ASNs in the feedback-learning network to include spike-based learning. Inspecting the learning rules (8)–(12) we see that four terms are involved in updating a synapse between a neuron i and j: the feedforward activation $x_i(t)$, the TD-error $\delta(t)$, the gradient of the transfer function $f'(a_i(t))$, and, for the hidden layer neurons j, the feedback activity from the winning action k, $z_k(t)$.

In the spiking-AuGMenT formulation, we use ASNs in both the forward and the feedback network, also while training the network. The feedforward and feedback activations $x_i(t)$ and $z_k(t)$ are both computed as a sum of spike-triggered kernels, corresponding to $S(t)$ in the ASN model. Reformulating CT-AuGMeNT, we denote the spiking neurons of spiking-AuGMenT with s and we use the same subscripts with the analog CT-AuGMenT. Instantaneous and transient units emit spikes to the regular and memory spiking neurons, respectively:

$$a^R_{\phi j}(t_s) = \sum\nolimits_{t_s} \sum\nolimits_i v^R_{ij} x_i(t_s) * \phi(t_s), \quad s^R_j(t_s) = f(a^R_{\phi j}(t_s)), \tag{18}$$

$$a^M_{\phi m}(t_s) = a^M_m(t_s - dt) + \sum\nolimits_{t_s} \sum\nolimits_l v^M_{lm} x'_l(t_s) * \phi(t_s), \quad s^M_m(t_s) = f(a^M_{\phi m}(t_s)), \tag{19}$$

where t_s is the time of outgoing spikes, f is the effective transfer function and $\phi(t)$ an exponential decay filter. As before, the Q-layer is fully connected to the association layer and the values are updated when there are input spikes:

$$q_k(t_s) = \sum_{t_s} \left(\sum_m w^M_{mk} \sigma^M_m(t_s) + \sum_j w^R_{jk} \sigma^R_j(t_s) \right). \tag{20}$$

Equivalently to the analog network, the Z-layer involves the action mechanism and determines the amount of inhibition an action receives. Note that now the transfer-function is implicit. The spiking neurons in the feedback network are defined as:

$$a^Z_{\phi k}(t_s) = \sum_{t_s} \sum_k z_k(t_s) * \phi(t_s), \tag{21}$$

$$s^R_{kj}(t_s) = f\big(\sum_{t_s} \sum_k w^R_{kj}(t_s) a^Z_{\phi k}(t_s)\big), \; s^M_{kj}(t_s) = f\big(\sum_{t_s} \sum_k w^M_{kj}(t_s) a^Z_{\phi k}(t_s)\big). \tag{22}$$

Equations (8)–(10) and (12) are reformulated accordingly, where we approximate the local gradient of the transfer-function as the derivative of the positive part of the *tanh*-function: $tanhP' = \max(0, 1 - \tanh^2)$ - while a rough approximation, we find this works well in practice. Tags between the association layer and the Q-layer are then defined as:

$$Tag_{jk}(t + dt) = -\frac{1}{\phi} Tag_{jk}(t) + dt[y_j(t) a^Z_{\phi k}(t_s))]. \tag{23}$$

For tags that are formed between the sensory layer and the association layer:

$$Tag_{ij}(t+dt) = -\frac{1}{\phi}Tag_{ij}(t) + dt[x_i(t_s)tanhP'(a^R_{\phi j}(t_s))s^R_{kj}(t_s)]. \tag{24}$$

$$\begin{aligned} sTrace_{lm}(t+dt) &= sTrace_{lm}(t) + dt[x'_l(t_s)], \\ Tag_{lm}(t+dt) &= -\tfrac{1}{\phi}Tag_{lm}(t) + dt[sTrace_{lm}(t)tanhP'(a^M_{\phi m}(t_s))s^M_{kj}(t)]. \end{aligned} \tag{25}$$

In the spike-based learning process the weights are updated again by (12), where the TD-error $\delta(t)$ is still an analog broadcasted signal.

In both tasks the initial weights are positive uniformly distributed, motivated by the rectified-positive nature of the spike-based feedback network (22) (Fig. 3a).

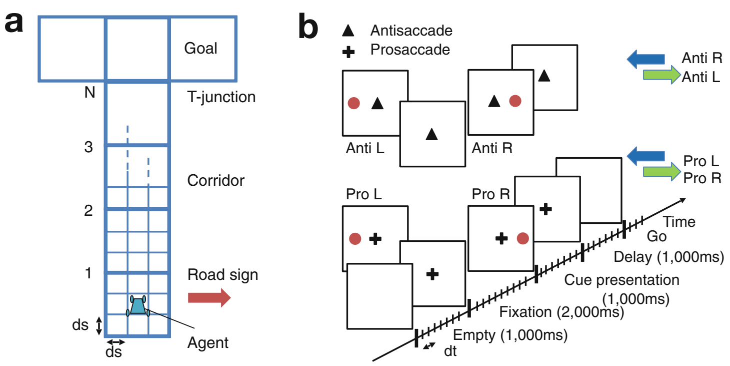


Fig. 4. Tasks. (a) T-Maze task, (b) Saccade-anti-Saccade task. See text for explanation.

Spike-Triggered Learning. In the spiking-AuGMenT formulation, each weight is updated every dt, even though the typical dynamics of the tasks have substantially longer temporal dynamics - milliseconds versus hundreds of milliseconds: a more sparse sampling approach to learning should suffice. Rather than fixed interval learning, we here propose to exploit the asynchronous nature of adaptive spike-coding: we only update the weights when a neuron receives or emits a spike (illustrated in Fig. 3b). The benefit of this sampling scheme is that with adaptive neural coding, the spike-rate increases there is a large change in signal, thus allowing for more and more precise sampling when needed. In more detail, whenever a neuron emits a spike we update the weights, otherwise the learning process pauses. Here, we denote with n the number of the current learning update. Hence, the rule for the update of the weights is:

$$\begin{aligned} v_{ij}(t_{s,n+1}) &= v_{ij}(t_{s,n}) + \delta t[\beta\delta(t)Tag_{ij}(t_{s,n})], \\ v_{lm}(t_{s,n+1}) &= v_{lm}(t_{s,n}) + \delta t[\beta\delta(t)Tag_{lm}(t_{s,n})], \\ w_{jk}(t_{s,n+1}) &= w_{jk}(t_{s,n}) + \delta t[\beta\delta(t)Tag_{jk}(t_{s,n})], \end{aligned} \tag{26}$$

where δt equals the time between two successive spikes: $\delta t = t_{s,n+1} - t_{s,n}$ (note that here each neuron updates only for its "own" spikes $t_{s,n}$).

5 Results

We demonstrate the spike-based CT-AuGMenT model of Fig. 1 on two working memory tasks: the T-Maze task from the machine learning literature [1,14] and the Saccade/Antisaccade task from the neuroscience literature (both as in [20]).

The **T-Maze** task is a working memory task where information that is presented at the start of the maze has to be maintained to make optimal decisions at the end of the corridor. The agent can choose actions to move in directions N, E, S, W; the corridor length N scales the task difficulty. The same details for corridor representation, reward and time-out conditions as in [20] were applied. For the simulations, we gave each network at most 10,000 trials to learn the task. Convergence was determined by checking at 90% optimal choices as in [20] for each condition. The parameters of the network for the T-Maze task are: $\beta = 0.02$, $\lambda = 0.3$, $\gamma = 0.9$, $\epsilon = 0.025$, $\tau = 0.5$ and corridor length $N = 10$. The ASNs use fixed values for $\theta_0 = 0.1$ and $\tau_\phi = 2.5$ ms. The network is updated at time increments of $dt = 0.01$, equivalent to 10 ms. The network consists of 24 neurons: a sensory layer with 9 input neurons (3 instantaneous and 6 transient units), an Association layer with 7 neurons (4 memory neurons and 3 regular neurons), and, matching the number of possible actions, both the output and the action layer have 4 neurons. Weights between the Sensory and Association and Q-layer are randomly initialized from the uniform distribution $U[0, 0.25]$.

In the **Saccade/Antisaccade (SaS)** task, the agent has to learn that the color of the fixation mark determines the strategy. Every trial started with an empty screen, shown for one second. Then a fixation mark was shown, either black or white, indicating that a pro- or anti-saccade was required. The model had to fixate within ten seconds, otherwise the trial was terminated without reward. If the model fixated for two consecutive seconds, we presented a cue on the left or the right side of the screen for one second and gave the fixation reward r_{fix}. This was followed by a memory delay of two seconds during which only the fixation point was visible. At the end of the memory delay the fixation mark turned off. To collect the final reward r_{fin} in the pro-saccade condition, the model had to make an eye-movement to the remembered location of the cue and to the opposite location on anti-saccade trials. The trial was aborted if the model failed to respond within eight seconds. The maximum number of trials the model is allowed to learn the task is set to 35,000. As to the implementation in [20], we kept the same temporal sequence of the events, and we updated the network at an increased rate of $dt = 0.01$ (corresponding to 10 ms per time step). The chosen parameters for the simulation are: $\beta = 0.01$, $\lambda = 0.2$, $\gamma = 0.9$, $\epsilon = 0.025$, $\tau = 0.5$, $\theta_0 = 0.1$ and $\tau_\phi = 2.5$ ms. The initialization of the weights is also uniformly distributed $U[0, 0.25]$. In this task the network is comprised of 26 neurons, with 12 neurons in the sensory layer (4 instantaneous and 8 transient units), 8 neurons in the Association layer (4 memory and 4 regular units) and both output and action layers have 3 neurons.

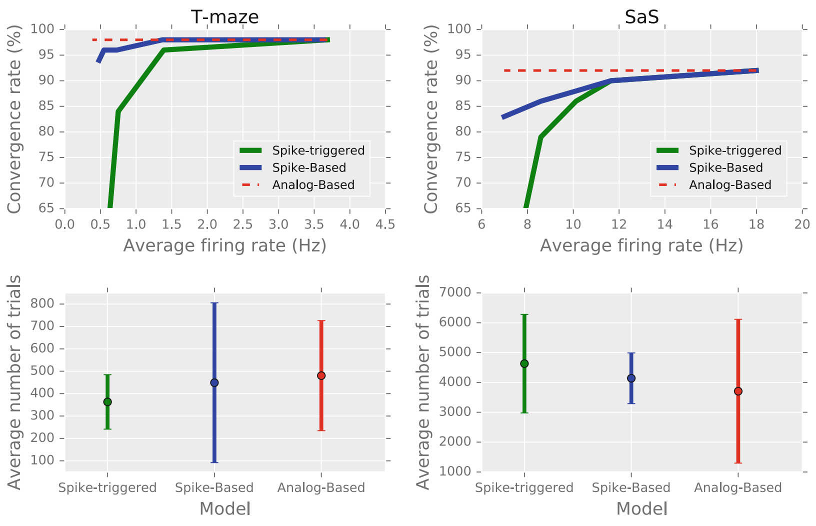


Fig. 5. *First row:* the convergence rate over the average firing rate (Hz) for the two tasks. In the T-Maze task we used $\tau_\gamma = [50, 150, 450, 1000, 1750]$ ms and $\tau_\eta = [150, 450, 1000, 1750, 2500]$ ms. In the SaS task we have $\tau_\gamma = 50$ ms fixed and $\tau_\eta = [100, 150, 200, 250, 300]$ ms. *Bottom row*: The average number of trials for each model and task for spiking network that match the analog network's convergence rate.

In both tasks, the spiking neuron time-constants τ_γ, τ_η are varied to generate spiking neurons that have varying asymptotic activation rates. We tested 50 randomly initialized networks for each set of τ_η and τ_γ. At the end of each learning phase we set $\beta = \epsilon = 0$ to validate the convergence.

We plot the results for both tasks in Fig. 5, both in terms of convergence rate of the networks (top row) and the number of trials required for learning the tasks. We find that both spiking methods, spike-based and spike-triggered CT-AuGMenT, are able to learn the tasks with convergence rates similar to that of CT-AuGMenT [19,20] and the analog rectified version (dashed line) for sufficiently high firing rates. We also compare the average number of trial needed for those spiking networks where the convergence rate matches the analog network (bottom row): we find that for all three learning models, the networks need a similar number of trials to converge. We note also that for both tasks, a majority of networks still converge even for very low average firing rates (<1 Hz for the T-Maze, <8 Hz for SaS).

6 Conclusion

We demonstrated how a continuous-time spiking neural network with working memory can be constructed with plausible spiking neuron models and plausible learning rules that uses on-policy reinforcement learning to learn hard cognitive tasks, a first such network to the best of our knowledge. These spiking neural networks learn the tasks equally fast as their analog counterparts, while needing very few spikes to both learn and carry out the neural computations. As such, this work can be considered an important milestone for creating efficient, sparsely-active and always-on neural networks, with promise for emerging neuromorphic paradigms like the Intel Loihi architecture [4].

Here, we focused on creating a spiking network to learn and compute Q-values, while using an analog action-selection system as we chose to focus here on the learning aspect of the tasks; we see no principled problem to create a spike-based version of this system. The spike-based transmission of potentially negative Q-values represents the greatest challenge, as we found that replacing the linear transfer functions in the Q-layer with half-sigmoid-like rectified functions - or the spiking equivalent, did not work; this is a challenge we presently tackling.

Compared to LSTM networks [9], the presented architecture lacks gating mechanisms and recurrence; LSTM-like gating however is notoriously hard to implement with spiking neurons in continuous time, and we find that for many tasks these structures however are not necessary. We will consider this in future research, incorporating for instance subtractive gating [3].

Acknowledgments. DZ is supported by NWO NAI project 656.000.005.

References

1. Bakker, B.: Reinforcement learning with long short-term memory. In: Dietterich, T., Becker, S., Ghahramani, Z. (eds.) NIPS 14, pp. 1475–1482 (2002)
2. Bohte, S.M.: Efficient spike-coding with multiplicative adaptation in a spike response model. In: NIPS 25, pp. 1844–1852 (2012)
3. Costa, R., Assael, I.A., Shillingford, B., de Freitas, N., Vogels, T.: Cortical microcircuits as gated-recurrent neural networks. In: NIPS 29, pp. 272–283 (2017)
4. Davies, M., Srinivasa, N., Lin, T.H., Chinya, G., Micro, Y.C.I.: Loihi: a neuromorphic manycore processor with on-chip learning. ieeexplore.ieee.org (2018)
5. Diehl, P., Neil, D., Binas, J., Cook, M., Liu, S.C., Pfeiffer, M.: Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In: IJCNN, pp. 1–8 (2015)
6. Gerstner, W., Kistler, W.: Spiking Neuron Models: Single Neurons, Populations, Plasticity. Cambridge University Press, Cambridge (2002)
7. Gilra, A., Gerstner, W.: Predicting non-linear dynamics by stable local learning in a recurrent spiking neural network. Elife **6**, e28295 (2017)
8. Gurney, K.N., Prescott, T.J., Redgrave, P.: A computational model of action selection in the basal ganglia. I. A new functional anatomy. Biol. Cybern. **84**, 401–410 (2001)
9. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
10. Lillicrap, T.P., Cownden, D., Tweed, D.B., Akerman, C.J.: Random synaptic feedback weights support error backpropagation for deep learning. Nat. Commun. **7**, 13276 (2016)
11. Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A.A., Veness, J.: Human-level control through deep reinforcement learning. Nature **518**, 529–533 (2015)

12. Niv, Y., Daw, N.D., Dayan, P.: Choice values. Nat. Neurosci. **9**(8), 987–988 (2006)
13. Roelfsema, P.R., van Ooyen, A.: Attention-gated reinforcement learning of internal representations for classification. Neural Comput. **17**(10), 2176–2214 (2005)
14. Rombouts, J., Bohte, S.M., Roelfsema, P.R.: Neurally plausible reinforcement learning of working memory tasks. In: NIPS 25, pp. 1880–1888 (2012)
15. Rombouts, J.O., Bohte, S.M., Roelfsema, P.R.: How attention can create synaptic tags for the learning of working memories in sequential tasks. PLoS Computat. Biol. **11**(3), e1004060 (2015)
16. Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L.: Mastering the game of Go with deep neural networks and tree search. Nature **529**(7587), 484–489 (2016)
17. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
18. Zambrano, D., Nusselder, R., Scholte, H.S., Bohte, S.: Efficient computation in adaptive artificial spiking neural networks. arXiv preprint arXiv:1710.04838 (2017)
19. Zambrano, D., Roelfsema, P., Bohté, S.: Learning continuous-time working memory tasks with on-policy neural reinforcement learning (2018, in preparation)
20. Zambrano, D., Roelfsema, P.R., Bohte, S.M.: Continuous-time on-policy neural reinforcement learning of working memory tasks. In: IJCNN 2015, April 2015