frontiers | Frontiers in Neuroscience

TYPE Original Research
PUBLISHED 03 February 2026
DOI 10.3389/fnins.2026.1738140

Check for updates

OPEN ACCESS

EDITED BY
Jiangrong Shen,
Xi'an Jiaotong University, China

REVIEWED BY
Zhaokun Zhou,
Peking University, China
Rong Xiao,
Sichuan University, China

*CORRESPONDENCE
Jaewook Kim
✉ jaewookk@kist.re.kr

RECEIVED 03 November 2025
REVISED 23 December 2025
ACCEPTED 12 January 2026
PUBLISHED 03 February 2026

CITATION
Shin D, Jo H, Jang H, Jeong YH, Jeong Y, Kwak JY, Park J, Lee S, Kim I, Park J-K, Park S, Jang HJ, Lee H-M and Kim J (2026) Spike-based Q-learning in a non-von Neumann architecture.
Front. Neurosci. 20:1738140.
doi: 10.3389/fnins.2026.1738140

COPYRIGHT
© 2026 Shin, Jo, Jang, Jeong, Jeong, Kwak, Park, Lee, Kim, Park, Park, Jang, Lee and Kim. This is an open-access article distributed under the terms of the Creative Commons Attribution License (CC BY). The use, distribution or reproduction in other forums is permitted, provided the original author(s) and the copyright owner(s) are credited and that the original publication in this journal is cited, in accordance with accepted academic practice. No use, distribution or reproduction is permitted which does not comply with these terms.

Spike-based Q-learning in a non-von Neumann architecture

Donghyuk Shin[1,2], Hyeongcheol Jo[1,2], Hyeseung Jang[1,2], Yoo Ho Jeong[2,3], YeonJoo Jeong[2], Joon Young Kwak[4], Jongkil Park[2], Suyoun Lee[2], Inho Kim[2], Jong-Keuk Park[2], Seongsik Park[2], Hyun Jae Jang[2], Hyung-Min Lee[1] and Jaewook Kim[2]*

[1]Korea University, Seoul, Republic of Korea, [2]Korea Institute of Science and Technology (KIST), Seoul, Republic of Korea, [3]LG Electronics Inc, Seoul, Republic of Korea, [4]Ewha Womans University, Seoul, Republic of Korea

Non-von Neumann architectures overcome the memory-compute separation of von Neumann systems by distributing computation and memory locally, thereby reducing data-transfer bottlenecks and power consumption. These features are particularly advantageous for reinforcement learning (RL) workloads that rely on frequent value-function updates across large state-action spaces. When combined with event-driven spiking neural networks (SNNs), non-von Neumann architectures can further improve overall computational efficiency by leveraging the sparse nature of spike-based processing. In this study, we propose a hardware-feasible SNN-based non-von Neumann architecture that performs Q-learning, one of the most widely known reinforcement learning algorithms. The proposed architecture maps states and actions to individual neurons using one-hot encoding and locally stores each state–action pair's *Q*-value in the corresponding synapse. To enable each synapse to update its local *Q*-value based on the next state maximum *Q* stored in other synapses, a neuron group connected through a lateral inhibition structure is employed to produce the maximum *Q*, which is then globally transmitted to all synapses. A delay circuit is also added to align the next-state and current-state values to ensure temporally consistent updates. Each synapse locally generates a learning selection signal and combines it with the globally transmitted signals to update only the target synapse. The proposed architecture was validated through simulations on the Cart-pole benchmark, showing stable learning performance under low-bit precision and achieving comparable accuracy to software-based Q-learning with sufficient bit precision.

KEYWORDS

non-von Neumann architecture, neuromorphic architecture, SNN, reinforcement learning, Q-learning, cart-pole

1 Introduction

Reinforcement learning (RL) provides a computational framework in which an agent learns optimal policies by interacting with the environment and receiving feedback in the form of rewards (Sutton and Barto, 2015). RL has been widely adopted in domains such as robotics, Internet of Things (IoT) systems, smart grid energy management, and communication systems, which are characterized by stringent power and latency constraints as well as the need to process large-scale streaming data efficiently (Spanò et al., 2019). To meet these requirements, researchers have focused on enhancing the

computational efficiency of RL algorithms. Parallel hardware acceleration platforms, including general-purpose GPUs (Tiwari et al., 2025), field-programmable gate arrays (FPGAs; Tran et al., 2022; Salomo et al., 2025), and custom accelerators (Spanò et al., 2019), have shown substantial improvements in processing speed. Nevertheless, such approaches still exhibit much lower energy efficiency than biological neural systems, highlighting a substantial gap between artificial and biological computation (Yamazaki et al., 2022).

As an alternative to close this gap, spiking neural networks (SNNs)—a bio-plausible third-generation neural model—have attracted considerable attention (Taherkhani et al., 2020; Mehonic and Kenyon, 2022; Kiselev et al., 2025). Due to their event-driven nature, SNNs remain largely inactive in the absence of spikes, thereby enabling highly energy-efficient computation. However, executing SNN-based algorithms on conventional von Neumann architectures still suffers from computational delays and energy overhead caused by sequential memory access and control logic bottlenecks (Haşegan et al., 2022; Liu and Pan, 2023; Liu et al., 2023; Siddique et al., 2023).

Neuromorphic processors such as Intel's Loihi (Davies et al., 2018), Stanford's Neurogrid (Benjamin et al., 2014), and IBM's TrueNorth (Akopyan et al., 2015) were developed to support spike-based computation. These architectures mitigate the structural bottlenecks of von Neumann systems and demonstrate the feasibility of large-scale spike-based processing with improved energy efficiency. Recent studies have successfully implemented RL algorithms, including Deep Q-Networks (DQN) and Deep Deterministic Policy Gradient (DDPG), on the Loihi platform, thus demonstrating their potential for real-time, low-power learning (Tang et al., 2020; Akl et al., 2021; Zanatta et al., 2023).

Despite remarkable progress in neuromorphic hardware, SNN processors are not yet fully non-von Neumann architectures due to programming requirements for general-purpose functionality. For example, Loihi employs programmable virtual synaptic connections to configure neural networks with reconfigurable connectivity. Once spikes are transmitted into a core, a sequence of operations within the core—including the identification of target neurons, retrieval and update of the associated neuronal and synaptic data from memory, and storage of results—causes computational latency. Parallelization across multiple cores can alleviate memory-access delays compared with conventional von Neumann architectures; however eliminating memory-search operations altogether would enable even greater energy efficiency.

In this work, we propose a non-von Neumann architecture that performs Q-learning—a well-established reinforcement learning algorithm—based on SNNs. States and actions are one-hot encoded into input and output neurons, respectively, and the synapses between them are hardwired with a fixed topology such that each synapse locally stores and updates the $Q(S, A)$ value through an up/down counter. This enables Q-table updates to be executed directly through spike events without requiring complex memory search or control logic, thereby reducing bottlenecks and improving energy efficiency.

A key challenge in this architecture is the distributed storage of Q-values across synapses, which complicates simultaneous access to both the $Q(S, A)$ of the current state and the maximum $Q(S', a)$ of the next state. Spatially, these values are stored in different local synapses and therefore are not directly accessible, while temporally, they do not coexist immediately after a state transition. Furthermore, because the target synapse for update is not predetermined, globally transmitting the maximum $Q(S', a)$ risks unintended simultaneous updates across multiple synapses.

This challenge is addressed by proposing three architectural mechanisms. First, a population of neurons is designed to compute the maximum $Q(S', a)$ in the next state through a lateral inhibition structure, and the resulting spikes are subsequently distributed globally. Second, spikes encoding the $Q(S, A)$ of the selected action in the current state are temporally delayed via a delay circuit to ensure their co-occurrence with the maximum $Q(S', a)$ spikes at the same time instance. Third, because spikes representing the current state and the selected action's Q-value are delivered simultaneously only to their corresponding synapses, their coincidence generates a selection signal that enables synapse-specific updates even in the presence of globally broadcast signals. These mechanisms enable each synapse to independently perform Q-learning updates without additional memory or address lookups.

The hardware feasibility of the proposed architecture is demonstrated through simulations in the cart-pole environment, a widely used reinforcement learning benchmark. The learning performance is further evaluated by varying the synaptic memory precision from 2 to 5 bits, allowing identification of the minimum precision required to sustain learning and the bit-width necessary to achieve performance comparable to conventional Q-learning. Such analysis provides practical insights into the trade-off between resource efficiency and learning performance in neuromorphic hardware implementations.

2 Background

Q-learning is a type of off-policy Temporal Difference (TD) learning, in which the value of the current state is updated using the estimated value of the next state. In off-policy learning, the behavior policy, which determines the agent's actions, is separated from the target policy that the agent aims to optimize. In Q-learning, the behavior policy is typically implemented using an epsilon-greedy policy, where an action is selected at random with probability ε, and the action that maximizes the reward is selected with probability $1 - \varepsilon$. The target policy, in contrast, follows a greedy policy that consistently selects the action associated with the highest Q-value.

The goal of Q-learning is to enable an agent to interact with its environment and learn an optimal policy that determines the best action A to take in each state S. The agent iteratively estimates the state–action value function $Q(S, A)$, which facilitates the selection of optimal actions with respect to the current state. The Q-learning update rule is given by

$$Q(S,A) \leftarrow Q(S,A) + \alpha(R + \gamma max_a Q(S', a) - Q(S,A)) \quad (1)$$

where $\alpha \in (0, 1]$ is the learning rate, which determines the extent to which newly obtained information overrides previously acquired estimates. R is the immediate reward received after taking action A in state S. The discount factor $\gamma \in [0, 1]$ determines the relative

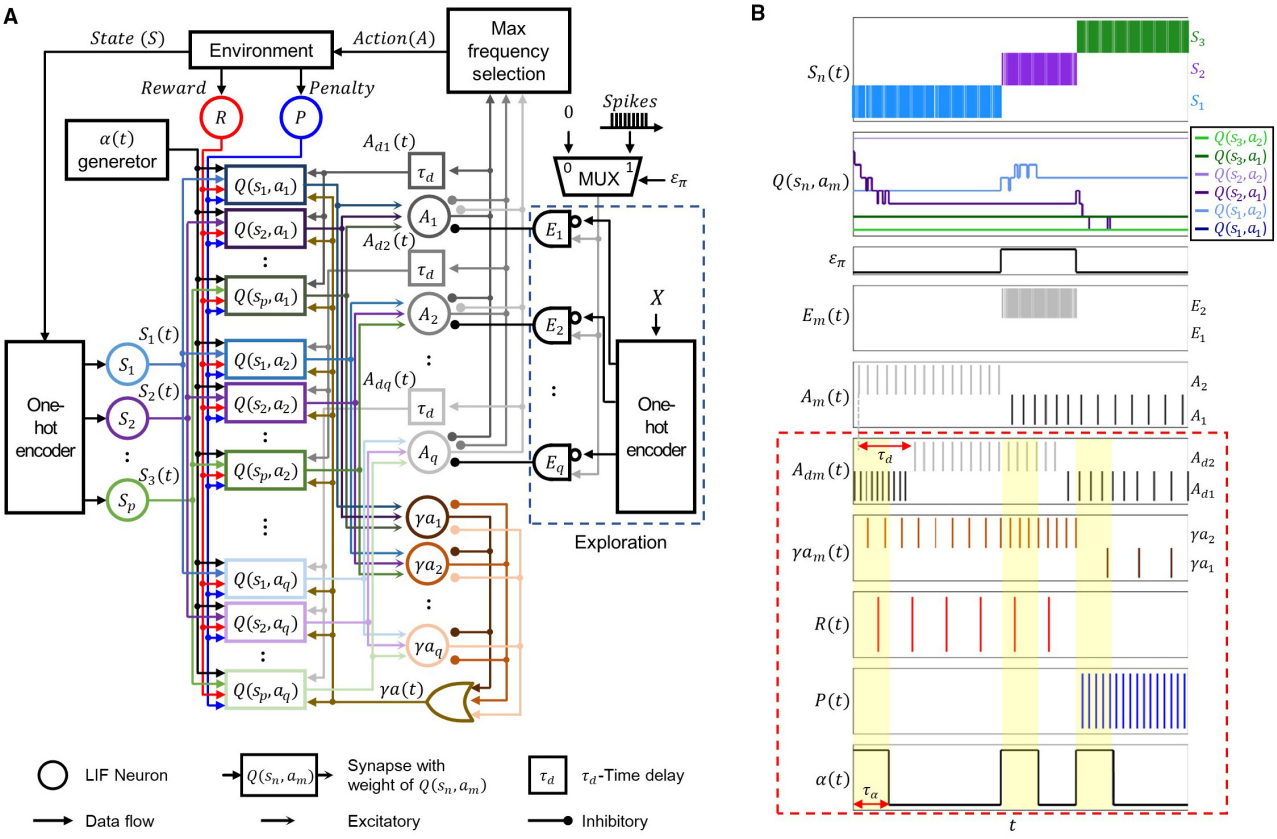


FIGURE 1
(A) Block diagram of the proposed non-von Neumann SNN architecture for Q-learning. **(B)** Operation waveforms of the proposed architecture for three states ($p = 3$) and two actions ($q = 2$). As the state transitions through s_1, s_2 and s_3, the state spikes $S_n(t)$ are generated. Depending on $S_n(t)$ and the exploration signal $E_m(t)$, which is randomly activated according to ε_π, $A_m(t)$ exhibits a firing frequency representing $Q(s_n, a_m)$. The delayed signal $A_{dm}(t)$ reflects $A_m(t)$ shifted by τ_d. Independent of $E_m(t)$, $\gamma a_m(t)$ generates spikes corresponding to the maximum $Q(s_n, a_m)$. Based on environmental feedback, either (t) or (t) fires, and upon each state transition, $\alpha(t)$ produces a pulse of duration τ_α.

importance of future rewards. The term $\max_a Q(S', a)$ represents the maximum estimated value of the next state S'. The Q-value is updated after the agent performs an action A in the current state S, interacts with the environment, and subsequently observes the next state S' together with the reward R.

3 Method

3.1 SNN architecture for Q-learning

3.1.1 State-action mapping and policy implementation

Figure 1A shows the proposed non-von Neumann architecture implementing SNN-based Q-learning, and Figure 1B illustrates the waveforms that represent the operation of architecture. States and actions are mapped to individual leaky integrate-and-fire (LIF) neurons (Abbott, 1999), enabling a direct mapping between state-action space and neural representation. The neurons representing states and those representing actions are fully connected, and the synapses between them correspond to the Q-table, with each synaptic weight encoding $Q(S, A)$. Each state neurons S_n ($n = 1, 2, \ldots, p$) represents one element of the state set $S = \{s_1, s_2, \cdots, s_p\}$, and each action neuron A_m ($m = 1, 2, \ldots, q$) represents one element of the action set $A = \{a_1, a_2, \cdots, a_q\}$.

The observed state S from the environment is one-hot encoded (Seger, 2018), producing a binary one-hot signal in which only the element corresponding to S is set to "1," whereas all others are set to "0." The resulting vector activates the corresponding state neuron S_n, which in turn generates spikes transmitted to the entire population of action neurons. Under the epsilon-greedy policy, action neuron A_m emits spikes with a firing rate proportional to $Q(s_n, a_m)$, determined by either exploitation or exploration.

In the proposed architecture, exploitation is implemented through a lateral inhibition structure, in which the outputs of the action neurons mutually suppress one another, allowing only the action neuron associated with the highest Q to become active. Exploration is implemented through the circuit shown in Figure 1A, where a discrete random variable X selects one element from the action set $A = \{a_1, a_2, \cdots, a_q\}$ with uniform probability whenever the state changes. The selected value is provided as input to a one-hot encoder, which converts it into a digital parallel signal. The encoder output is then processed through an inverter, and the inverted signal is combined with the spikes generated by the MUX

via an AND gate, resulting in either spikes or 0. These combined spikes suppress the action neurons before lateral inhibition takes effect, thereby allowing only the neuron corresponding to X to remain active and emit spikes proportional to Q.

The balance between exploitation and exploration is determined by the discrete random variable ε_π, which takes the value 0 with probability $1-\varepsilon$ and 1 with probability ε. When $\varepsilon_\pi = 0$, the MUX output is 0, and the architecture operates in exploitation mode without suppression of the action neurons by the AND gates. Conversely, when $\varepsilon_\pi = 1$, the MUX output generates spikes that pass through the AND gates and suppresses all but one action neuron, thereby enabling exploration. The outputs of the action neurons are subsequently transmitted to a selection module that identifies the action neuron with the highest firing frequency and delivers the corresponding action A to the environment.

Figure 1B illustrates the spiking activity of the state neurons in response to state transitions and the spiking of the action neurons as determined by the value of ε_π for $p = 3$ and $q = 2$. The state changes asynchronously in the order of s_1, s_2, s_3, causing the corresponding S_1, S_2, and S_3 neurons to fire sequentially. After each state transition, $Q(s_n, a_m)$ is immediately updated and details on this Q update process are provided in Section 3.2. For s_1 and s_3, where exploitation is applied, the A_2 and A_1 neurons fire according to the highest Q, $Q(s_1, a_2)$ and $Q(s_3, a_1)$, respectively. In contrast, for s_2, where exploration is applied, the A_1 neuron fires despite $Q(s_2, a_2)$ being higher than $Q(s_2, a_1)$, due to the suppression of the A_2 neuron by the E_2 spikes.

3.1.2 Spike encoding for Q-learning updates

To adapt Q-learning updates for SNN operation, the elements of Equation 1, which include the reward R, $\gamma max_a Q(S', a)$, and $Q(S, A)$, are encoded as spike signals whose firing frequencies are proportional to their respective values and delivered to the target synapses. The learning rate α is represented as a pulse whose width is proportional to the corresponding value and is transmitted to all synapses. The encoding and delivery of these transformed signals are illustrated in the red dashed box in Figure 1B.

Spike signals with firing frequencies proportional to $Q(S, A)$ and $\max_a Q(S', a)$ can be generated by introducing LIF neurons driven by these terms. In the proposed architecture, however, the firing frequency of each action neuron is inherently proportional to $Q(S, A)$. Therefore, spike signals representing $Q(S, A)$ can be obtained directly from the existing action neurons without the need for additional circuitry (Figure 1A).

When exploration is applied under the epsilon-greedy policy, spike signals with firing frequencies proportional to $\max_a Q(S', a)$ is not obtainable from the action neurons. To address this limitation, the proposed architecture incorporates additional γa_m neurons (Figure 1A). These neurons share the same synaptic connections as the action neurons but implement only the lateral inhibition structure associated with exploitation. The scaling factor γ is determined by adjusting the thresholds of the γa_m neurons, and their firing frequencies vary in proportion to γ.

For the Q-learning update, both the Q-value of the current state and that of the next state are required to be simultaneously available. However, because only the current Q-value is stored in synapses, the Q-value of the next state becomes available only after the state transition. To resolve this issue, the proposed architecture incorporates a delay mechanism that enables the coexistence of the current and next Q-values within the next state by delaying the A_m spikes corresponding to $Q(S, A)$. Specifically, the spike signals $A_m(t)$, which represent to $Q(S, A)$, are delayed by a fixed time interval τ_d to generate $A_{dm}(t)$. This delayed signal ensures that, during the next state, spikes representing $Q(S, A)$ remain available for a duration of τ_d.

The outputs of each A_m neuron are delayed individually, such that spikes corresponding to $Q(S, A)$ can be delivered to the synapses of the same A_m neuron during the next state, thereby enabling the Q-learning update. The outputs of the γa neurons are combined using an OR gate to form a single $\gamma(t)$ signal, which is then delivered to all synapses.

In Figure 1B, spikes corresponding to $Q(S, A)$ and spikes corresponding to $\gamma max_a Q(S', a)$ coexist for a duration of τ_d immediately following a state transition. For example, after the transition from s_1 to s_2, A_{d2} spikes appear with a firing frequency proportional to $Q(s_1, a_2)$, while, within the same interval, γa_2 spikes emerge with a frequency proportional to $Q(s_2, a_2)$.

The reward signal R, positive for rewards and negative for penalties, is converted into spikes without sign information by introducing two additional neurons (Figure 1A): an R neuron for rewards and a P neuron for penalties. For each state, when a reward occurs, only the P neuron emits spikes with a frequency proportional to the reward magnitude, whereas when a penalty occurs, only the P neuron emits spikes with a frequency proportional to the penalty magnitude. The spikes from theses neurons are delivered to all synapses to drive the Q-learning update.

The spike signals corresponding to the terms in Equation 1 coexist only during a limited interval τ_d after a state transition, which defines the effective update window in the next state. In the proposed architecture, the learning rate α is implemented by an α generator that produces a pulse of width τ_α, proportional to α and bounded by τ_d. This pulse is triggered at each state transition and only spikes occurring within the τ_α window contribute to the Q-learning updates. A smaller τ_α results in fewer spikes being involved in the computation. As illustrated in Figure 1B, when $\tau_\alpha < \tau_d$, only R, P, A_{dm} and γa_m spikes within the τ_α window are utilized for learning.

3.1.3 Spike-based synaptic update circuit for Q-learning

As shown in Figure 1A, $A_{dm}(t)$, $R(t)$, $P(t)$, and $\gamma a(t)$ are delivered globally to all synapses. $S_n(t)$ is transmitted only to the synapses connected to the specific state neuron S_n, while $A_{dm}(t)$ is transmitted exclusively to the synapses connected to the selected action neuron A_m. Consequently, both signals are simultaneously present only at synapses where the current state and the currently selected action are jointly represented. The proposed architecture exploits this structural feature to generate a selection signal based on these two inputs, which in turn determines the Q-learning update.

10.3389/fnins.2026.1738140

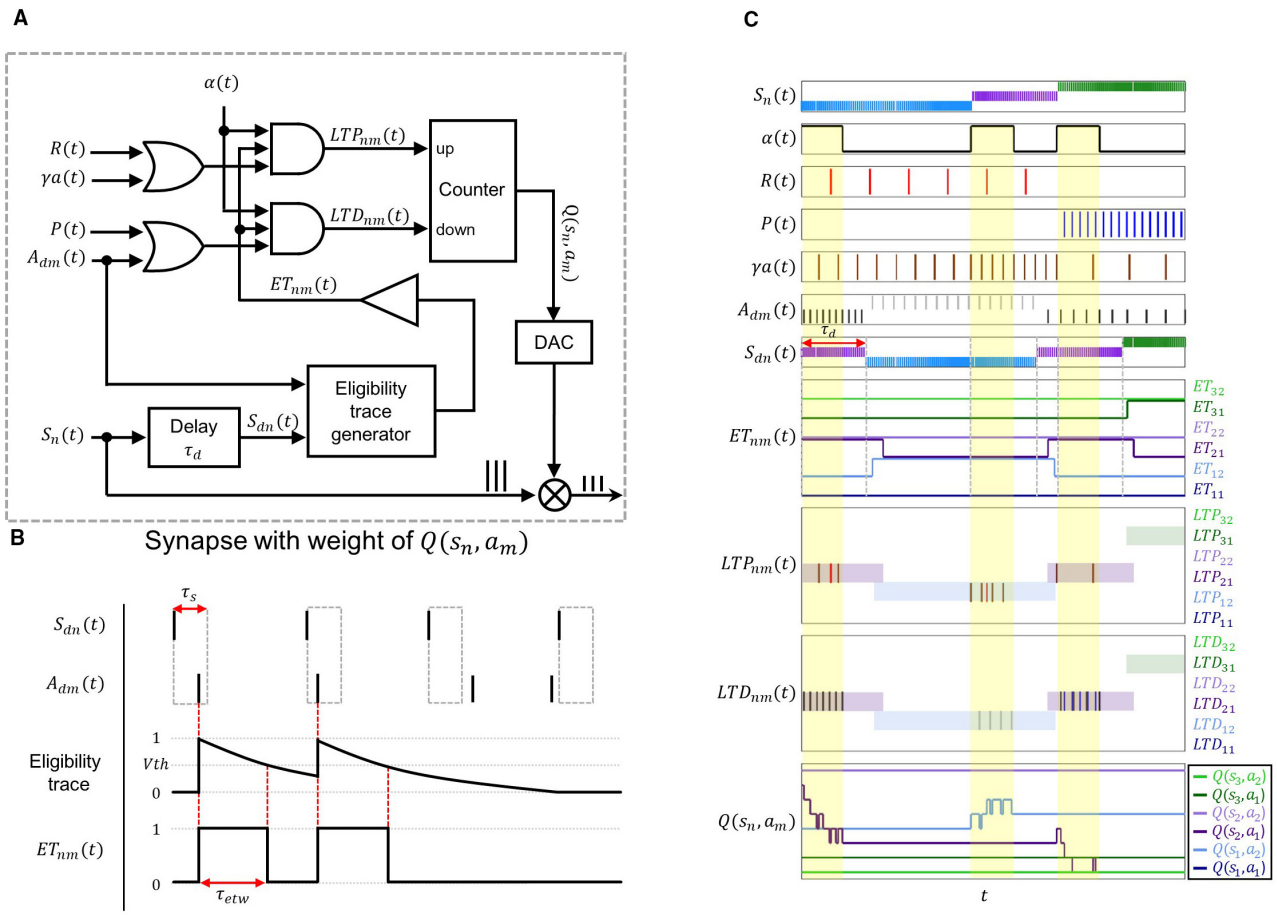


FIGURE 2
(A) Block diagram of a synaptic circuit performing local updates of the weight corresponding to $Q(s_n, a_m)$, where the delayed state and delayed action spikes generate an eligibility trace that is combined with update-related inputs to produce LTP_{nm}/LTD_{nm} spikes driving the counter-based Q-value update. **(B)** Operation waveforms of the eligibility trace generator and the waveform conversion of the trace using a buffer. The eligibility trace is generated when an A_{dm} spike occurs within τ_s after an S_{dn} spike, and this trace is converted into the ET_{nm} pulse of duration τ_{etw} through a buffer with a threshold V_{th}. **(C)** Operation waveforms illustrating the $Q(s_n, a_m)$ update process based on signals transmitted to and generated within the synaptic block. $ET_{nm}(t)$ pulses are generated when the delayed state signal $S_{dn}(t)$, obtained by shifting $S_n(t)$ by τ_d, coincides with $A_{dm}(t)$. When the yellow-shaded $\alpha(t)$ pulse overlaps with $ET_{nm}(t)$ pulses, $LTP_{nm}(t)$ spikes are induced by (t) and $\gamma a(t)$, whereas $LTD_{nm}(t)$ spikes are induced by $P(t)$ and $A_{dm}(t)$. Each $LTP_{nm}(t)$ and $LTD_{nm}(t)$ spike updates $Q(s_n, a_m)$ by a single step.

Figure 2A shows the block diagram of an individual synapse in the proposed architecture, which performs the computations required for Q-learning update and stores the Q-values. As Q-learning updates occur during the τ_d period of the next state, the S_n spikes are delayed by τ_d to remain valid within this interval. After the occurrence of S_{dn} spikes, the subsequent arrival of A_{dm} spikes generate the eligibility trace.

The eligibility trace generator can be realized using a capacitor–MOSFET structure, in which capacitors integrate incoming spikes and discharge gradually through leakage, while a MOSFET gates the signal according to the resulting voltage. This circuit configuration produces a decaying trace that represents synaptic eligibility (Wijekoon and Dudek, 2011). The resulting trace is converted by a buffer into an ET_{nm} pulse of duration τ_{etw} (Figure 2B), defining the time window in which learning is valid. As illustrated in Figure 2C, $ET_{nm}(t)$ is generated at the synapse corresponding to $Q(S, A)$ during the τ_d period of the next state. After the transition from s_1 to s_2, the S_{d1} spikes from the previous state s_1 and the A_{d2} spikes generate an ET_{12} pulse that remains HIGH for the duration of τ_d, enabling only $Q(s_1, a_2)$ to be updated in state s_2.

The up/down counter is employed to store $Q(s_n, a_m)$ values and to update them using spike-based signals (Figure 2A). The input spikes are generated by classifying the signals in Equation 1 into those that increase $Q(s_n, a_m)$ and those that decrease it, and combining each group with logic gates. Specifically, (t) and $\gamma(t)$ are grouped for potentiation, and $P(t)$ and $A_{dm}(t)$ are grouped for depression, with each pair combined through OR gates. The outputs of the OR gates are subsequently gated by (t) and $ET_{nm}(t)$ using AND operations, producing $LTP_{nm}(t)$ and $LTD_{nm}(t)$ signals.

The up/down counter receives LTP_{nm} spikes at its up input, increasing $Q(s_n, a_m)$ by one count per spike, and LTD_{nm} spikes at its down input, decreasing $Q(s_n, a_m)$ by one count per spike. Figure 2C shows the synaptic updates of $Q(s_n, a_m)$ driven by spikes in the proposed architecture. Within the time window where the α pulse and the ET_{nm} pulse coexist, LTP_{nm} spikes

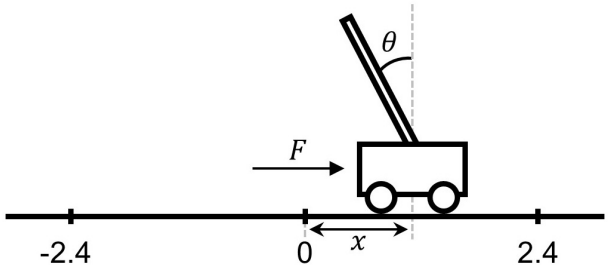


FIGURE 3
Cart-pole game environment.

are generated from the combination of R spikes and γa spikes, while LTD_{nm} spikes arise from the combination of P spikes and A_{dm} spikes. Each occurrence of an LTP_{nm} spike results in a real-time increase in $Q(s_n, a_m)$, whereas each LTD_{nm} spike results in a real-time decrease.

3.2 Cart-pole task environment

The cart-pole task, illustrated in Figure 3, is a standard benchmark in reinforcement learning where a force is applied to a cart along the x-axis on a flat surface with the goal of maintaining the pole balanced on the cart (Geva and Sitte, 1993). In this study, simulations were conducted using the cart-pole environment provided in the Reinforcement Learning Toolbox of MATLAB. Each episode was initialized with the cart positioned at the origin and the pole in an upright orientation. At every 20 ms time step, a force of either +10 N or 10 N was applied to the cart. An episode terminates in failure if the cart position exceeds ±2.4 units from the origin or if the pole angle exceeds ±12 °. Conversely, an episode is considered successful if the pole remained balanced within these bounds for 4 s.

The state variables of the cart-pole environment are the cart position x, cart velocity $\dot{x}$, pole angle θ, and pole angular velocity $\dot{\theta}$. These variables were quantized as follows:

$$x : (-2.4, 2.4)\,;\ \dot{x} : (-\infty, \infty)\,;$$
$$\theta : (-12,\ -0.1]\,,\ (-0.1, -0.01],\ (-0.01,\ 0],\ (0,\ 0.01],$$
$$(0.01,\ 0.1],\ (0.1,\ 12)\,;$$
$$\dot{\theta} : (-\infty,\ -0.87],\ (-0.87,\ 0.87],\ (0.87, \infty).$$

The state set S consists of 19 elements, comprising 18 four-dimensional tuples from the combinations of the state variables and one failure state of the cart-pole task. Within the action set $A = \{-10\,\text{N},\ 10\,\text{N}\}$, the proposed architecture contained 38 synapses encoding the corresponding $Q(s_n, a_m)$ values. For non-failure states s_1- s_{18}, a reward of +1 is assigned, whereas for the failure state s_{19}, a penalty of −8 is applied. The parameter ε in the epsilon-greedy policy, which determines the probability of exploitation and exploration, was initialized at 1 and decays by a factor of 0.7 across episodes.

TABLE 1 Model parameters used in the simulation of the proposed architecture.

Parameter	Value
Bit-width	3-bit
S_n freq (Hz)	10k
τ_d (ms)	5
τ_α (ms)	5
Reward freq (Hz)	205
Penalty freq (Hz)	1,700
τ_{etw} (ms)	14

4 Experiments & results

To evaluate the operation of the proposed non-von Neumann architecture in a hardware-oriented context, a high-level simulation model was implemented in MATLAB and interfaced with the cart-pole environment. All simulations were performed on a workstation with an Intel(R) Core™ i7-8700 CPU @ 3.20 GHz and 16 GB of RAM.

In the simulations, the model parameters were set as follows: the learning rate $\alpha = 1$, the discount factor $\gamma = 0.99$, a counter bit-width of 3 bits, a reward of +1, and a penalty of −8. The firing frequency of the state neurons was fixed at 10 kHz, whereas the action neurons fired at frequencies ranging from 201 to 1,610 Hz depending on the Q-values stored in their corresponding synapses. The eligibility trace window τ_{etw} was set to 14 ms to ensure that, at the lowest action neuron frequency of 201 Hz, the trace generated by an A_{dm} spike persisted in the buffer until the next spike arrived. A detailed summary of the simulation parameters is provided in Table 1.

Based on these parameters, we evaluated the proposed architecture in the cart-pole environment across 100 episodes. Figure 4 shows the simulated waveforms of Episodes 1, 30, and 100. The signals $R(t)$, $P(t)$, $\gamma a(t)$, and $A_{dm}(t)$ denote spike trains over time, whereas $Q(s_n, a_1)$ and $Q(s_n, a_2)$ represent the corresponding Q-values, updated in time and quantized to 3 bits. The colors of the $Q(s_n, a_1)$ and $Q(s_n, a_2)$ traces are matched to those of the corresponding S_n spikes to indicate correspondence.

In episode 1, the $Q(s_n, a_m)$ values were initialized to their maximum. Since all $Q(s_n, a_m)$ values were identical at the start, most of them changed only slightly during learning. However, once the state transitioned to the failure state s_{19}, $Q(s_{18}, a_1)$ decreased sharply in response to the P spikes, leading to the termination of the episode.

In episode 30, the initial Q-values reflected the learning accumulated from previous episodes. Within the green-shaded interval between 0.96 s and 1.44 s, the state transitioned from s_9 to s_2, with the action A_1 selected in both states. At s_9, the $Q(s_9, a_1)$, shown by the thick blue trace, corresponds to $Q(S, A)$, whereas at s_2, the $Q(s_2, a_1)$, shown by the thick orange line, corresponds to $\max_a Q(S', a)$.The Q-learning update defined in Equation 1 was executed, causing $Q(s_9, a_1)$ to decrease by four steps. Similar to episode 1, episode 30 also terminated when the state reached the failure state s_{19} at 1.44 s.

10.3389/fnins.2026.1738140

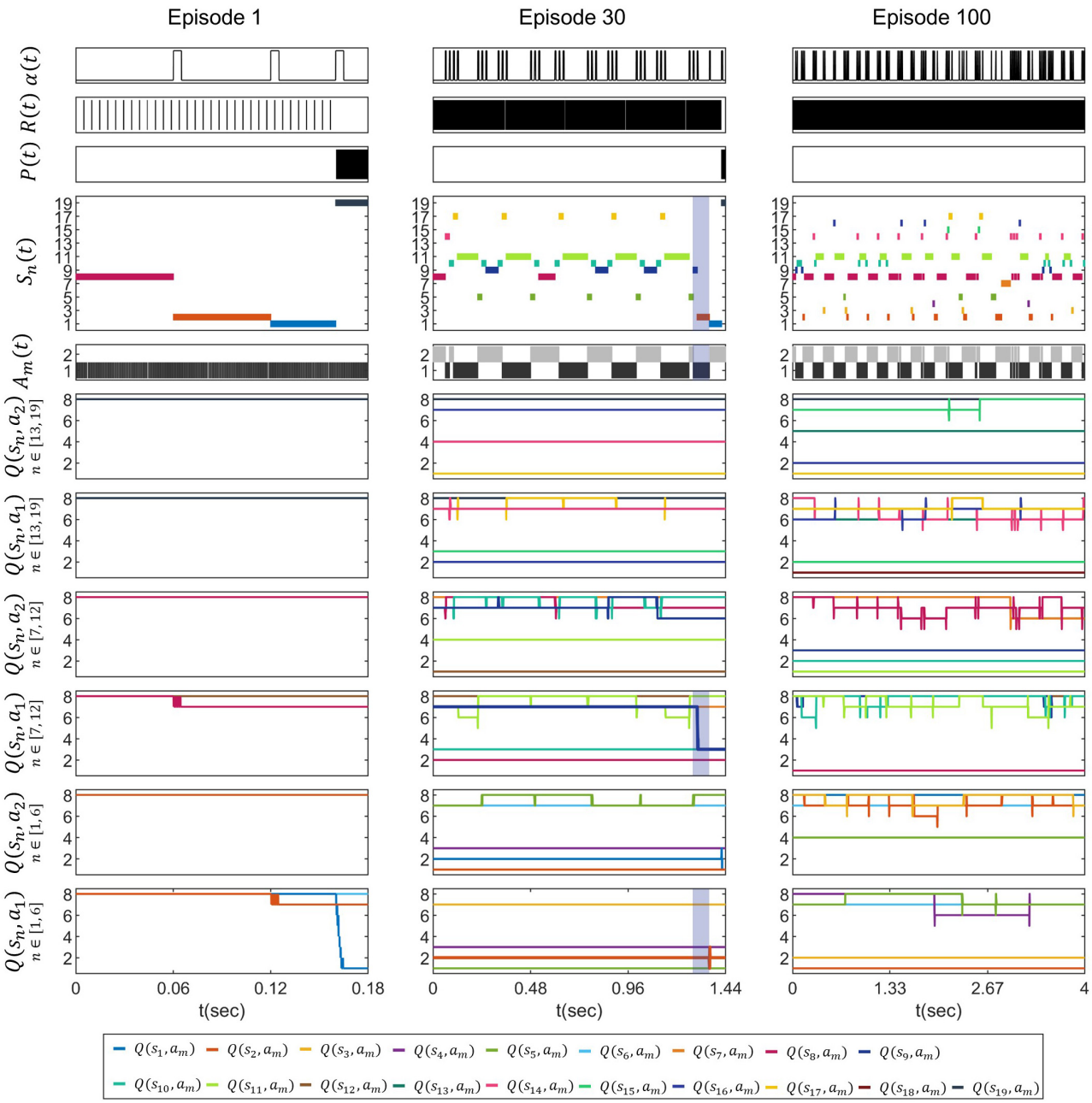


FIGURE 4
Simulation results of the cart-pole task for episodes 1, 30, and 100, showing failures at 0.18 s and 1.94 s in episode 1 and 30, and successful balance at 4 s in episode 100. Each panel shows the learning rate pulse (t), the reward spikes (t), the penalty spikes $P(t)$, the state spikes S_n (t) for n = 1, 2, ..., 19, the action spikes $A_m(t)$ for $m = 1$, 2, and the 3-bit quantized Q-values Q (s_n, a_m), represented using integer levels from 1 to 8. The Q-value trajectories are shown in separate Q (s_n, a_1) and Q (s_n, a_2) panels, with each panel corresponding to a different subset of states (s_1–s_6, s_7–s_{12}, and s_{13}–s_{19}).

In episode 100, the simulation terminated successfully after maintaining balance for the full 4 s without entering the failure state s_{19}. The $Q(s_n, a_m)$ had stabilized and, apart from minor deviations of approximately one step following updates, remain largely unchanged from their prior values.

The performance of the proposed architecture was evaluated by averaging scores every 20 episodes across 10 independent simulation runs. The score increased by 1 for every 20 ms in which the pole remained balanced, reaching a maximum of 200 when balance was maintained for 4 s.

The red traces in Figure 5A show the average score per 20 episodes for each of the 10 simulations conducted with a 3-bit counter, while the black trace shows the mean of these averages across simulations. Although individual runs vary due to exploration governed by the epsilon-greedy policy, the results indicate that the average score reaches 200 within 100 episodes.

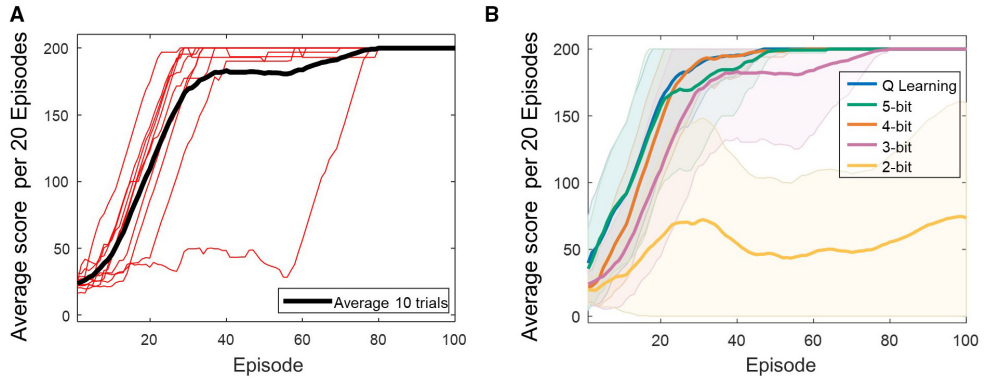


FIGURE 5
(A) Learning curves obtained using a 3-bit counter in the proposed architecture. Red lines indicate the average score per 20 episodes for each of the 10 trials, and the black line shows the overall mean. **(B)** Comparison of average score per 20 episodes across different counter bit-widths: 5-bit (green), 4-bit (orange), 3-bit (pink), and 2-bit (yellow), and standard Q-learning (blue). The solid lines show the average score per 20 episodes over 10 trials, and the shaded area represent the standard deviation.

TABLE 2 Parameters for different counter bit widths in the proposed architecture.

Parameter	Value			
Bit-width	2-bit	3-bit	4-bit	5-bit
S_n freq (Hz)	10k	10k	20k	40k
τ_d (ms)	2	5	8	10
τ_α (ms)	2	5	8	10
Reward freq (Hz)	505	205	127	105
Penalty freq (Hz)	2,200	1,700	2,050	3,250
τ_{etw} (ms)	17	14	11	9

Figure 5B compares the average score per 20 episodes across 10 simulations with $\alpha = 1$ and $\gamma = 0.99$, under the counter bit-widths of 2, 3, 4, and 5, as well as conventional Q-learning without bit limitations. The experimental parameters for each counter bit configuration are summarized in Table 2. In this experiment, the parameters for each bit-width configuration were selected to ensure stable operation of the architecture. The reward was fixed at the minimum unit of +1, while the penalty was set to the maximum negative value representable by each bit-width. Furthermore, the frequencies of the reward and penalty signals were adjusted so that the number of spikes associated with each value was appropriately reflected within the maximum valid time window τ_d.

With the 2-bit counter (yellow trace), the cart-pole task failed as the average score does not reach 200. The 3-bit counter (pink trace) achieved success approximately 40 episodes later than conventional Q-learning (blue trace), whereas the 4-bit (orange trace) and 5-bit (green trace) counters reached an average score of 200 within about 50 episodes, comparable to Q-learning. These results demonstrate that the proposed architecture can successfully solve the cart-pole task with a 3-bit counter, while performance comparable to Q-learning is obtained with a 4-bit counter.

TABLE 3 One-way ANOVA across quantization levels.

Source	SS	*df*	MS	*F*-value	*p*-value
Quantization level	160,810	4	40,204	60.0544	<0.0001
Error	30,125	45	669.45		
Total	190,940	49	8		

TABLE 4 Tukey's HSD *post-hoc* comparisons between Q-learning and models with different bit-widths.

Comparison	Mean diff	95%	*p*-value
Q-learning−5-bit	−0.0360	[−32.9148, 32.8428]	1.0000
Q-learning−4-bit	0.2700	[−32.6088, 33.1488]	1.0000
Q-learning−3-bit	5.7375	[−27.1413, 38.6163]	0.9874
Q-learning−2-bit	143.1677	[110.2889, 176.0465]	<0.0001

The performance graph in Figure 5B, generated using the parameters listed in Table 2, was analyzed using a one-way analysis of variance (ANOVA), and the results are summarized in Table 3. A statistically significant effect of quantization level on performance was observed [$_{(4,45)} = 60.0544$, p-value < 0.0001], encompassing unquantized Q-learning and 2–5-bit representations. Subsequently, Tukey's honestly significant difference (HSD) *post-hoc* tests were performed to compare Q-learning with each bit-width and the results are presented in Table 4. *Post-hoc* analyses revealed no significant differences between Q-learning and the 5-bit, 4-bit, or 3-bit models (all p-values ≥ 0.987). In contrast, the 2-bit condition showed significantly lower performance compared with Q-learning (p-value < 0.0001).

5 Discussion

In this study, we proposed a non-von Neumann SNN architecture specialized for the Q-learning algorithm. The

proposed system employs a hard-wired connectivity with a fixed network topology, in which each synapse stores a single Q-value, thereby reducing memory-access overhead through localized storage. This architectural approach contrasts with general-purpose neuromorphic processors such as Intel's Loihi, which adopt reconfigurable neural connectivity to support various network topologies but typically involve centralized or shared memory access, potentially leading to memory-access bottlenecks within the core. In this context, this work emphasizes algorithm-hardware co-optimization rather than hardware reconfigurability, suggesting a promising direction for improving computational efficiency. This approach aligns with prior studies emphasizing the need for co-design across multiple levels of neuromorphic systems—including hardware, circuits, algorithms, and applications (Schuman et al., 2022)—and suggests the potential of algorithm-centered hardware specialization as a direction for future neuromorphic hardware development.

These architectural differences are reflected in the energy efficiency and area characteristics. In term of energy efficiency, the synaptic weights in Loihi are stored in SRAM, and each spike is processed through AER address decoding, synapse selection, memory access, and a read–modify–write update, with the spike delivered as a packet across the on-chip network. While this packet-based event-driven approach is highly efficient for sparse activity, the dynamic power consumed per spike can increase as spike events are transmitted in packet form. In contrast, in the proposed architecture, each Q-value is stored in a local counter and spikes are routed directly through fixed wiring, thereby avoiding packet conversion and address decoding and reducing the amount of data movement and the activation of update-related circuitry.

In terms of area, Loihi is designed such that the neurons and synapses within each core share a common computation and learning engine, whereas in the proposed architecture, dedicated processing units and local learning circuits are assigned to each neuron and synapse block. As a result, Loihi can achieve a relatively higher neuron and synapse density per unit area. However, when the full system architecture is considered, Loihi includes additional blocks such as the network on chip (NoC), AER interface logic for packet processing, and routers, which contribute non-negligibly to the overall chip area. By comparison, although separate blocks for the NoC and packet-based routing are not required in the proposed architecture, additional area overhead arises from the increased wiring needed for the fixed connectivity between neuron and synapse blocks. The practical impact of these factors in implementation will require further examination and careful evaluation.

Another notable aspect of the proposed architecture is its alignment with biological learning processes observed in the brain. In the proposed system, distributed computation occurs locally at each synapse, global reward signals are broadcast throughout the network, and synapse-specific learning is achieved through local signal generation—analogous to the interplay between global modulatory signals and local synaptic events in the brain. In the brain, slow global signals such as hormones or neuromodulators regulate long-term learning, while local spike interactions at specific synapses drive plasticity (Brzosko et al., 2019). Similarly, the proposed architecture globally propagates both the reward and $\max_a Q(S', a)$, and generates local selection signals through the coincidence of pre- and post-synaptic spikes corresponding to state–action pairs. Moreover, the delay mechanism introduced to address temporal mismatches aligns with biological timing characteristics. Neural systems exhibit axonal conduction delays (Madadi Asl et al., 2017), synaptic transmission delays, and recurrent-circuit delays, all of which play crucial roles in learning mechanisms such as spike-timing-dependent plasticity (STDP). These similarities suggest that the proposed non-von Neumann architecture captures key functional aspects of biological learning mechanisms.

From a hardware perspective, this study demonstrated that a minimal 3-bit precision up/down counter used as a synaptic memory was sufficient to complete the cart-pole simulation within 100 episodes, confirming the feasibility of low-precision memory in practical learning. As the architecture scales, the number of synapses ($p \times q$) grows much faster than the number of neurons ($p + 2q$), making synaptic memory bit width and area efficiency critical constraints in hardware design. Therefore, the finding that stable learning can be achieved with as few as 3 bits supports the practical feasibility of implementing the proposed architecture on neuromorphic hardware.

Beyond precision considerations, it is also important to assess whether the proposed architecture remains robust when scaled to larger network sizes. In conventional von Neumann systems, Q-values are stored in centralized memory, requiring frequent memory accesses and substantial data movement during learning. Consequently, memory bottlenecks have been a major limitation when such systems are scaled. In contrast, in the proposed architecture, Q-values are stored in local counters within each synapse block, and learning is carried out in parallel across synapses, so that memory-related bottlenecks do not arise structurally during scaling.

A remaining concern in large-scale expansion is whether propagation delays along long signal routes could introduce timing mismatches in learning. In the proposed architecture, learning is based on counting spikes within an α pulse of duration τ_α, with a maximum timing tolerance defined by τ_d. Global update-related signals—such as $S_n(t)$, $A_{dm}(t)$, $\gamma a(t)$, $R(t)$, and $P(t)$—are routed across synapse blocks through wires of varying lengths. Differences in wire lengths can introduce arrival-time variations, which may affect the number of spikes captured within the α pulse and lead to non-uniform Q-updates across the network. In the presented 3-bit implementation, update-related signals operate at a maximum frequency of 10 kHz, such that a 1% timing variation corresponds to approximately 1 μs. In 16–22 nm technology nodes, the reported propagation delay is about 2 ns per millimeter [International Technology Roadmap for Semiconductors (ITRS), 2007]. At this rate, a delay of 1 μs would accumulate only over wire lengths exceeding approximately 555 mm, which is far beyond the dimensions of a typical single chip. Even in multi-chip board-level configurations, substantial margin therefore remains before routing-induced delays would meaningfully affect learning behavior.

In addition to these hardware-level considerations, large-scale Q-learning presents challenges, particularly in terms of slower

convergence and reduced generalization when the state–action space becomes very large. As the number of states increases, experience becomes sparsely distributed across the space, reducing opportunities for repeated correction of specific situations. This sparsity slows learning and can lead to generalization errors in which the agent assigns inaccurate Q-values to insufficiently explored states. As the dimensionality of the environment grows, these challenges become more severe, often requiring substantially more interactions to achieve stable learning outcomes. In future work, large-scale simulations may be used to evaluate the impact of update sparsity on performance, and concepts inspired by similarity-based update approaches (Rosenfeld et al., 2017) may be incorporated to ensure that related state–action pairs reflect the most recent environmental information even under sparse updates. Additionally, the proposed architecture incorporating these approaches may also be implemented on neuromorphic hardware.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

DS: Writing – original draft, Conceptualization, Data curation, Formal analysis, Methodology, Software, Writing – review & editing, Validation, Visualization. HyeoJ: Data curation, Methodology, Writing – original draft. HyesJ: Formal analysis, Writing – original draft, Visualization. YHJ: Software, Validation, Writing – original draft. YJ: Investigation, Writing – review & editing, Validation. JYK: Writing – review & editing, Investigation, Methodology. JP: Funding acquisition, Writing – review & editing, Investigation. SL: Writing – review & editing, Funding acquisition, Resources. IK: Investigation, Writing – review & editing. J-KP: Writing – review & editing, Investigation. SP: Validation, Writing – review & editing, Software. HyuJ: Writing – review & editing, Software, Validation. H-ML: Writing – review & editing, Investigation, Supervision. JK: Conceptualization, Funding acquisition, Investigation, Methodology, Project administration, Resources, Supervision, Validation, Writing – original draft, Writing – review & editing.

Funding

The author(s) declared that financial support was received for this work and/or its publication. This work was supported in part by the Korea Institute of Science and Technology (KIST) under Grants 2E33560 and 2E33721, in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korea government (Ministry of Science and ICT, MSIT) (RS-2025-02217259), and in part by the National R&D Program through the National Research Foundation of Korea (NRF) funded by MSIT (2021M3F3A2A01037808).

Acknowledgments

We thank Sungsoo Han and Youngwoong Song for their technical assistance and valuable discussions.

Conflict of interest

YHJ was employed by LG Electronics Inc.

The remaining author(s) declared that this work was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Generative AI statement

The author(s) declared that generative AI was not used in the creation of this manuscript.

Any alternative text (alt text) provided alongside figures in this article has been generated by Frontiers with the support of artificial intelligence and reasonable efforts have been made to ensure accuracy, including review by the authors wherever possible. If you identify any issues, please contact us.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Abbott, L. F. (1999). Lapicque's introduction of the integrate-and-fire model neuron. *Brain Res. Bull.* 50, 303–304. doi: 10.1016/S0361-9230(99)00161-6

Akl, M., Sandamirskaya, Y., Walter, F., and Knoll, A. (2021). "Porting deep spiking Q-networks to neuromorphic chip Loihi," in *ACM International Conference Proceeding Series* (New York, NY: Association for Computing Machinery). doi: 10.1145/3477145.3477159

Akopyan, F., Sawada, J., Cassidy, A., Alvarez-Icaza, R., Arthur, J., Merolla, P., et al. (2015). TrueNorth: design and tool flow of a 65 mW 1 million neuron programmable neurosynaptic chip. *IEEE Trans. Comput. Aided Design Integrated Circuits Syst.* 34, 1537–1557. doi: 10.1109/TCAD.2015.2474396

Benjamin, B. V., Gao, P., McQuinn, E., Choudhary, S., Chandrasekaran, A. R., Bussat, J. M., et al. (2014). Neurogrid: a mixed-analog-digital multichip system for large-scale neural simulations. *Proc. IEEE* 102, 699–716. doi: 10.1109/JPROC.2014.2313565

Brzosko, Z., Mierau, S. B., and Paulsen, O. (2019). Neuromodulation of spike-timing-dependent plasticity: past, present, and future. *Neuron* 103, 563–581. doi: 10.1016/j.neuron.2019.05.041

Davies, M., Srinivasa, N., Lin, T.-H., Chinya, G., Cao, Y., Choday, H., et al. (2018). *Loihi: A Neuromorphic Manycore Processor with On-Chip Learning*. Available online at: www.computer.org/micro (Accessed January 15, 2026).

Geva, S., and Sitte, J. (1993). A cartpole experiment benchmark for trainable controllers. *IEEE Control Syst. Magaz.* 13, 40–51. doi: 10.1109/37.236324

Haşegan, D., Deible, M., Earl, C., D'Onofrio, D., Hazan, H., Anwar, H., et al. (2022). Training spiking neuronal networks to perform motor control using reinforcement and evolutionary learning. *Front. Comput. Neurosci.* 16:1017284. doi: 10.3389/fncom.2022.1017284

International Technology Roadmap for Semiconductors (ITRS) (2007). *International Technology Roadmap for Semiconductors: Interconnect*. San Jose, CA: ITRS.

Kiselev, M., Ivanitsky, A., and Larionov, D. (2025). A purely spiking approach to reinforcement learning. *Cogn. Syst. Res.* 89:101317. doi: 10.1016/j.cogsys.2024.101317

Liu, G., Deng, W., Xie, X., Huang, L., and Tang, H. (2023). Human-level control through directly trained deep spiking Q-networks. *IEEE Trans. Cybern.* 53, 7187–7198. doi: 10.1109/TCYB.2022.3198259

Liu, Y., and Pan, W. (2023). Spiking neural-networks-based data-driven control. *Electronics* 12:310. doi: 10.3390/electronics12020310

Madadi Asl, M., Valizadeh, A., and Tass, P. A. (2017). Dendritic and axonal propagation delays determine emergent structures of neuronal networks with plastic synapses. *Sci. Rep.* 7:39682. doi: 10.1038/srep39682

Mehonic, A., and Kenyon, A. J. (2022). Brain-inspired computing needs a master plan. *Nature* 604, 255–260. doi: 10.1038/s41586-021-04362-w

Rosenfeld, A., Taylor, M. E., and Kraus, S. (2017). "Speeding up tabular reinforcement learning using state-action similarities," in *Proceedings of the 16th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2017)*, eds. E. Durfee, M. Winikoff, K. Larson, and S. Das (Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems), 1722–1724.

Salomo, Y., Syafalni, I., Sutisna, N., and Adiono, T. (2025). Hardware-software stitching algorithm in lightweight Q-learning system on chip (SoC) for shortest path optimization. *IEEE Access* 13, 105044–105062. doi: 10.1109/ACCESS.2025.3578681

Schuman, C. D., Kulkarni, S. R., Parsa, M., Mitchell, J. P., Date, P., and Kay, B. (2022). Opportunities for neuromorphic computing algorithms and applications. *Nat. Comput. Sci.* 2, 10–19. doi: 10.1038/s43588-021-00184-y

Seger, C. (2018). *An Investigation of Categorical Variable Encoding Techniques in Machine Learning: Binary Versus One-hot and Feature Hashing* (Master's thesis). KTH Royal Institute of Technology School of Electrical Engineering and Computer Science, Stockholm, Sweden.

Siddique, A., Vai, M. I., and Pun, S. H. (2023). A low cost neuromorphic learning engine based on a high performance supervised SNN learning algorithm. *Sci. Rep.* 13:6280. doi: 10.1038/s41598-023-32120-7

Spanò, S., Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Giardino, D., Matta, M., et al. (2019). An efficient hardware implementation of reinforcement learning: the q-learning algorithm. *IEEE Access* 7, 186340–186351. doi: 10.1109/ACCESS.2019.2961174

Sutton, R. S., and Barto, A. G. (2015). *Reinforcement Learning: An Introduction, 2nd Edn*. Cambridge: MIT Press.

Taherkhani, A., Belatreche, A., Li, Y., Cosma, G., Maguire, L. P., and McGinnity, T. M. (2020). A review of learning in biologically plausible spiking neural networks. *Neural Netw.* 122, 253–272. doi: 10.1016/j.neunet.2019.09.036

Tang, G., Kumar, N., Yoo, R., and Michmizos, K. P. (2020). *Deep Reinforcement Learning with Population-Coded Spiking Neural Network for Continuous Control*. Available online at: https://github.com/combra-lab/pop-spiking-deep-rl (Accessed January 15, 2026).

Tiwari, G., Nakhate, S., Pathak, A., Jain, A., and Penurkar, S. (2025). "Hardware accelerators for deep learning applications," in *2025 IEEE International Students' Conference on Electrical, Electronics and Computer Science, SCEECS 2025* (New York, NY: Institute of Electrical and Electronics Engineers Inc.). doi: 10.1109/SCEECS64059.2025.10940371

Tran, D. D., Le, T. T., Duong, M. T., Pham, M. Q., and Nguyen, M. S. (2022). "FPGA design for deep Q-network: a case study in Cartpole environment," in *2022 International Conference on Multimedia Analysis and Pattern Recognition, MAPR 2022 – Proceedings* (New York, NY: Institute of Electrical and Electronics Engineers Inc.). doi: 10.1109/MAPR56351.2022.9925007

Wijekoon, J. H. B., and Dudek, P. (2011). "Analogue CMOS circuit implementation of a dopamine modulated synapse," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS 2011)* (New York, NY: Institute of Electrical and Electronics Engineers Inc.), 877–880. doi: 10.1109/ISCAS.2011.5937706

Yamazaki, K., Vo-Ho, V. K., Bulsara, D., and Le, N. (2022). Spiking neural networks and their applications: a review. *Brain Sci.* 12:863. doi: 10.3390/brainsci12070863

Zanatta, L., Di Mauro, A., Barchi, F., Bartolini, A., Benini, L., and Acquaviva, A. (2023). Directly-trained spiking neural networks for deep reinforcement learning: energy efficient implementation of event-based obstacle avoidance on a neuromorphic accelerator. *Neurocomputing* 562:126885. doi: 10.1016/j.neucom.2023.126885