SEARCHABILITY OF EVOLUTIONARY SPIKING CONTROLLERS UNDER HARD ENERGY TRUNCATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Training spiking neural networks (SNNs) is important for energy-efficient and biologically plausible computation. However, existing BPTT-based methods suffer from hard truncation, which breaks the eligibility trace and renders gradients unusable for propagation. Meanwhile, STDP-based methods avoid this gradient issue but provide no explicit task-driven direction, leaving the search trajectory underdefined. To address this issue, we propose MED-SNN, an energy-constrained evolutionary framework that uses energy itself as the constraint driving search. In this way, energy shapes the search direction, while a gradient-free genetic algorithm performs optimization, thereby avoiding the gradient failure caused by hard truncation. The framework is gradient-free and directly encodes topology, weights, and metabolic parameters. It employs hard energy truncation—halting rollouts at the death threshold so that overspending trajectories leave the admissible set—and introduces inter-generational budget matching, which starts loose and progressively tightens toward the target energy ledger to preserve early selection signals. On CartPole-v1, MED-SNN remains searchable under training-time hard death (death rate 0.77). Hard-off: 409.9 ± 108.9 steps, 45% solve. Hard-on survival 0%. On the schedule table, matched-flat records hard-off 441.3 steps, 60% solve, and 100% hard-on survival; staged, linear, and cosine solve at 45%, 25%, and 45%. We claim training-time searchability under hard death.

1 INTRODUCTION

SNNs process information through discrete spike events and temporal membrane potential states. Their computation aligns with biological neural systems in terms of temporal coding and sparse firing, reflecting the low-power characteristics of biological neural computation. This makes SNNs a strong candidate for low-power edge intelligence. However, training SNNs effectively has long faced a fundamental difficulty: the threshold decision for spike generation is a non-differentiable step function, which cannot directly provide gradient information for backpropagation.

Existing SNN training methods form two groups. Surrogate-gradient BPTT approximates the non-differentiable spike and is a strong control baseline (Neftci et al., 2019; Wu et al., 2018); SpikingJelly packages this family, and BPTT-unroll denotes full temporal unrolling (Fang et al., 2023). Hard truncation of a rollout breaks eligibility traces, so the gradient stops (Bellec et al., 2020). STDP-style local plasticity avoids that derivative, yet leaves the task objective implicit (Hasegan et al., 2022). Neither family usually imposes at train time a hard energy constraint that would also apply at deployment. MED-SNN therefore lets remaining energy drive search, and a genetic algorithm updates gene-encoded controllers without differentiating through spikes (Stanley & Miikkulainen, 2002; Schuman et al., 2020).

We recast training as evolutionary search under that constraint. Topology, weights, and metabolic parameters live in one genome; crossover and mutation explore the mixed discrete–continuous space. Each controller is an individual with a genotype, an energy account, and a lifetime: spikes spend energy, task-correct behavior injects energy, and exhaustion ends the episode. Survivors enter a mating pool by weighted fitness and produce the next generation. We instantiate this as MED-SNN. The framework also exposes developmental gating as an optional encoding interface; the main experi-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

ments isolate metabolism and evolution, and do not activate gating. Hard truncation is a feasibility boundary of evaluation. A generational budget then loosens early, to keep diversity, and tightens later, toward the target constraint.

On CartPole-v1 (five seeds) and Acrobot-v1 (seed 42), with comparisons, ablations, a same-protocol e-prop control, and a boundary probe, MED-SNN learns task-capable policies under the hard energy constraint. Matching the budget to the horizon is required for search to proceed; the shape of the decay schedule leaves final performance unchanged. Under the same truncation protocol, local gradient training loses trainability, while MED-SNN remains searchable.

Our contributions are summarized as follows:

Hard energy truncation: We elevate energy feasibility from a weighted term in the fitness function to a feasibility boundary in evaluation. Any trajectory below the threshold terminates immediately, so overspending trajectories leave the admissible set.

Inter-generational budget scheduling: We schedule the energy budget across generations, starting loose to preserve early selection signals and tightening toward the target ledger.

Gene-encoded SNN search under the ledger: We encode topology, weights, and metabolic parameters as a genome and search with a genetic algorithm under the hard energy constraint.

2 Related Work

2.1 Surrogate-gradient SNN training

Surrogate-gradient training keeps a true spike on the forward pass and, on the backward pass, replaces the spike derivative with that of a smooth function (Neftci et al., 2019). Spatio-temporal backpropagation follows this path; BPTT-unroll stores the full time graph (Wu et al., 2018). SpikingJelly packages the same idea for actor–critic control, including CartPole (Fang et al., 2023). Lu and Sengupta pre-train with conversion and then fine-tune with differential evolution (Lu & Sengupta, 2022). e-prop writes credit into eligibility traces—local records of recent activity—and does not store the unrolled graph (Bellec et al., 2020).

However, these methods assume that an episode runs to completion before any credit is assigned. Under a hard energy ledger, the episode terminates as soon as energy reaches zero. Gradients from BPTT and eligibility traces in e-prop are cut off at that point, so credit cannot accumulate over the full trajectory. This assumption fails when energy feasibility is enforced during training.

2.2 Gradient-free evolutionary methods

Neuroevolution searches mixed discrete–continuous controllers without differentiating the spike. NEAT aligns crossover across changing topologies with historical markings and speciation, so structure and weights move in one search (Stanley & Miikkulainen, 2002). EONS puts thresholds, leaks, delays, and weights in a single SNN genome (Schuman et al., 2020). Later encodings add Shapley-weighted EONS fitness (Rodriquez & Ghawaly, 2026), neuron-centric Hebbian parameters (Ferigo et al., 2024), and multi-objective small-world and criticality scores (Pan et al., 2024). EVOL reaches 500 CartPole steps with a simple evolutionary loop and is the unconstrained gradient-free baseline here (Hasegan et al., 2022).

However, these methods rank individuals by task return and provide no explicit direction beyond that scalar. EONS subtracts a size penalty, which lowers the rank of costly rollouts without steering the population away from overspending; EVOL carries no metabolic account. Curriculum learning schedules task difficulty over training time (Bengio et al., 2009), while the schedule here adjusts the strictness of the energy budget across generations—one changes the task distribution, the other the feasibility boundary of evaluation. EONS encodes thresholds, leaks, delays, and weights on a fixed topology. The genome here also carries topology, connectivity, and metabolic parameters, coupling the search to the energy ledger. Under a hard energy ledger, no mechanism directs the population toward feasible trajectories before exhaustion removes them from the pool.

2.3 METABOLIC MODELS AND ARTIFICIAL LIFE

Neuron-level metabolic models couple firing to an energy-like store. Jaras et al. add a local ATP-dependent hyperpolarization term to energy-dependent LIF neurons and obtain attractors that differ from unconstrained dynamics (Jaras et al., 2025). The mAdExp model puts energy and spike limits into single-neuron equations (Fardet & Levina, 2020). PolyWorld places metabolism, survival, Hebbian learning, and sexual reproduction in one 2D ecology (Yaeger, 1994). The Werld artificial-life system revisits that ecology with NEAT-ANN controllers. NAGI pairs NEAT with STDP for RL in changing environments (Pontes-Filho et al., 2022). NDP and LNDP grow topology within a lifetime through GNN messages that split neurons and add edges (Najarro et al., 2023).

However, these systems act at the single-neuron scale or with continuous-valued agents. PolyWorld already stops an agent when its store is empty. The open question is whether a gene-encoded SNN controller remains searchable when a counting ledger truncates the control episode and gates reproduction.

In summary, existing work covers BPTT-based gradient training, STDP-based local plasticity, and metabolic modeling. BPTT-based methods break eligibility traces under hard truncation, while STDP-based methods lack an explicit task-driven direction. When evaluation stops at energy exhaustion, both lines of work face a setting they were not designed for: the episode ends before the trajectory completes. We evaluate gene-encoded SNNs in this setting, where hard truncation removes overspending trajectories from the admissible set and the search must still find viable policies.

3 METHODS

This section describes how the hard energy ledger is implemented in evaluation. In biological systems, learning proceeds without an explicit error sent backward through the network. An individual acts in its environment. Each spike draws on a metabolic store, and the outcome of action decides whether resources can refill that store. Only individuals that keep the store solvent survive and pass structure to offspring. When the store is empty, behavior stops. This observation motivates a search in which energy feasibility constrains who may continue, and the weighted fitness on the completed trajectory supplies the selection signal.

We therefore study gradient-free training of an SNN, a network that emits a spike when its membrane reaches threshold and then resets. At every environment step the controller maps the current state to an action; the ledger credits a survival allowance and a task injection, and debits spike, synapse, leak, existence, and connection costs. Fitness ranks genomes by a weighted mix of survival steps, remaining energy, metabolic cost, and genome size

$$F = w_{\text{steps}} \cdot \text{steps} + w_{\text{energy}} \cdot E_{\text{final}} - w_{\text{cost}} \cdot C_{\text{total}} - \alpha \cdot \|\text{genome}\|_0, \tag{1}$$

and a trajectory that reaches $E \leq 0$ ends. The unit of search is a genome that stores topology, connectivity, and metabolic parameters. A genetic algorithm updates that genome by crossover and mutation, without differentiating through spikes. We instantiate this as MED-SNN, which asks whether a gene-encoded spiking controller remains searchable when evaluation itself runs a hard counting ledger.

Figure 1 shows one evaluation pass at three scales: generation, lifetime, and substep.

Formally, let the environment state space be $\mathcal{S}$, the action space be $\mathcal{A}$, and the energy account be $E \in \mathbb{R}_+$. A controller π is decoded from genome G, and its behavior trajectory $\tau = s_0 a_0 s_1 a_1 \dots s_T$ is determined jointly by environment transitions and controller decisions. At each time step, after the controller executes an action, the energy account updates as

$$E_{t+1} = E_t + R_{\text{task}} \cdot \mathbf{1}_{\text{task}} - C_\pi(s_t). \tag{2}$$

When $E_{t+1} \leq E_{\text{death}}$, the trajectory is judged infeasible and evaluation terminates immediately. The task performance of a feasible trajectory is quantified by its survival steps T.

3.1 THREE TIME SCALES

We use three nested time scales to organize individual evaluation and the genetic process. The standard operations of genetic algorithms—selection, crossover, and mutation—all occur at the gen-

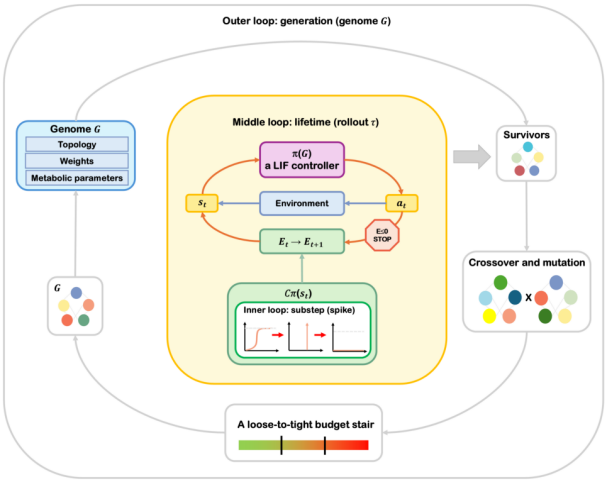


Figure 1: Nested evaluation of MED-SNN. The outer loop carries genome G (topology, weights, and metabolic parameters) to a LIF controller, then to survivors, crossover, and mutation. The middle loop is one rollout: the environment updates $E_t \to E_{t+1}$, and $E \le 0$ stops the trajectory. The inner loop charges the spike cost $C_\pi(s_t)$. A loose-to-tight budget stair runs across early, mid, and late generations.

erational scale. However, when training SNNs under hard energy constraints, the generational scale alone cannot address two preceding issues. First, how spike activity generates energy consumption requires tracking the fine-grained spiking details of the SNN at the substep scale. Second, judging an individual as infeasible due to energy depletion during its lifetime requires evaluating the cumulative energy balance over the complete trajectory. Therefore, three scales are needed to correspond to three aspects of hard-constrained training.

Substep scale. We use the LIF neuron model to account for the metabolic cost of spike emission and synaptic transmission. The LIF model balances computational cost against biological detail, and its membrane leakage mechanism corresponds to resting-state energy consumption. This scale addresses where energy comes from. Without substep-scale tracking, the metabolic cost of spikes cannot be quantified.

Lifetime scale. The energy account accumulates the energy balance at each environment step. When the account falls below a threshold, the trajectory terminates. This scale addresses when an individual is judged as infeasible. Without lifetime-scale truncation, the hard constraint cannot take effect during evaluation.

Generational scale. Genomes are updated through selection, crossover, and mutation, while the metabolic budget progresses from loose to tight. This scale addresses how the population accumulates experience across generations. Without generational-scale selection, the outcome of a single lifetime cannot translate into adjustments in the search direction. Substep and lifetime scales alone settle cost and truncation for one rollout; they do not update the genome pool or advance the loose-to-tight budget path. The generational scale supplies both of those operations.

Merging the substep and lifetime scales would settle the energy account once per environment step. A spike-heavy controller could then spend the same account as a sparse one over the same number of steps, and the search would lose the signal that distinguishes efficient controllers from wasteful ones. These three scales are nested. They transform training from minimizing a loss function on a fixed topology into screening for feasible strategies under hard energy constraints through genetic selection. The following two subsections describe the two core mechanisms.

3.2 Energy Hard-Truncation

Inspired by energy-dependent neural dynamics and the artificial-life survival loop, this work elevates metabolic constraints from dynamical models to a hard truncation mechanism usable in evaluation.

The energy hard-truncation mechanism responds to the observation in biological systems that energy depletion can directly terminate behavior. In biological neurons, action-potential generation and synaptic transmission constitute the main sources of energy consumption. The ATP consumed by the $\mathrm{Na}^+/\mathrm{K}^+$ pump to restore ion gradients after each spike, together with the energy demand of the action potential itself, forms the core of the spike cost. When local ATP supply is insufficient, pump efficiency decreases, the resting membrane potential shifts, firing frequency declines until it stops, and energy depletion means the physical termination of behavior.

Our training evaluation process reproduces this causal chain. Each time an individual completes one step of interaction in the environment, the system settles its energy account. Income items consist of a fixed survival allowance R_{surv} at each step and an additional energy injection R_{task} upon task achievement. Expenditure items include metabolic costs for spike emission, synaptic transmission, membrane leakage, neuron existence, and connection maintenance. Each cost is weighted by a fixed coefficient and included in the total consumption. The energy update equation is

$$\begin{aligned} E_{t+1} = E_t + R_{\mathrm{surv}} + R_{\mathrm{task}} \cdot \mathbf{1}_{\mathrm{task},t} \\ - (c_{\mathrm{spike}} s_t + c_{\mathrm{syn}} s_t^{\mathrm{syn}} + c_{\mathrm{leak}} s_t^{\mathrm{leak}} \\ + c_{\mathrm{exist}} n_{\mathrm{neurons}} + c_{\mathrm{conn}} n_{\mathrm{synapses}}). \end{aligned} \tag{3}$$

R_{surv} in equation 3 keeps randomly initialized individuals alive long enough to emit a selection signal. The default late $R_{\mathrm{surv}} = 2$ keeps the late training ledger tight, so hard-on survival near zero under that setting reports the same tight contract at test rather than a reason to revise the default. Raising late R_{surv} as in the survive-boost ablation is a separate reporting condition for deployment-style survival. R_{task} converts task performance into survival resources, linking task completion to survival. $c_{\mathrm{spike}} s_t$ binds spike emission to energy consumption. The remaining four terms correspond to synaptic vesicle release, resting-potential maintenance, structural existence, and connection maintenance.

When $E_{t+1} \leq E_{\mathrm{death}}$, the individual is marked as dead, the trajectory terminates immediately, and no further environmental interaction occurs. The default value of E_{death} is zero: the energy account does not allow overdraft. Ranking then follows equation 1. Here steps is the number of survival steps, E_{final} is the remaining energy at termination, C_{total} is the cumulative metabolic cost, and $\|\mathrm{genome}\|_0$ is the genome size. Deceased individuals have their fitness halved before they enter the mating pool. The mating pool is the upper half of the fitness-sorted population with fitness greater than zero. Whether an individual exceeds the energy budget thus becomes the criterion for whether the trajectory can continue to exist.

3.3 Inter-generational budget scheduling

Curriculum learning schedules task difficulty over training time. The schedule here adjusts the strictness of the energy budget across generations. The two differ in what they vary: one changes the task distribution, the other changes the feasibility boundary of evaluation. The energy hard-truncation mechanism addresses how to express a hard budget, but it introduces a new difficulty. If the budget is as tight in generation 0 as it is in the later stages of training, randomly initialized individuals will exhaust their energy before producing any useful behavior. The selection signal will disappear, and the population will collapse. Therefore, searchability under hard constraints itself becomes a problem that needs to be solved. The budget cannot be static; it must be gradually tightened as the population evolves.

To address this problem, this work adjusts the energy budget in stages across generations. Young organisms have relatively low energy consumption, high plasticity, and looser metabolic constraints on behavioral exploration. In adulthood, energy constraints become tighter and behavioral patterns stabilize. We adopt a three-stage schedule that progressively adjusts the initial energy budget, survival allowance, task allowance, and spike cost coefficients according to generation number. We map this process to the generational scale, where the three training boundaries correspond to exploration, transition, and fine-tuning. In the early phase, the budget is kept loose so that individuals engaged in random exploration obtain a sufficient survival window to produce preliminary task signals. In the middle phase, the budget is gradually tightened to start selecting strategies with better energy efficiency. In the late phase, the budget approaches the target hard constraint strength, retaining only strategies that complete the task with limited energy.

The core principle of this strategy is that the budget must follow a transition path from loose to tight across generations. If the same intensity were maintained throughout, the search would become extinct in the early stages. The slope of this transition path can vary, but the existence of the path itself is a necessary condition.

4 EXPERIMENTS

We evaluate MED-SNN on CartPole-v1, Acrobot-v1, and discrete-torque Pendulum. CartPole-1000 and MountainCar-v0 are boundary probes. Baselines are unconstrained EVOL (Hasegan et al., 2022), STDP-RL from the same source, an EONS-proxy size penalty (Schuman et al., 2020), STBP (Wu et al., 2018), BPTT-unroll (Neftci et al., 2019; Wu et al., 2018), SpikingJelly RL (Fang et al., 2023), e-prop without hard death (Bellec et al., 2020), and CMA-ES (Hansen & Ostermeier, 2001). Reported metrics are mean test score over 20 episodes, sample standard deviation, solve rate, wall-clock seconds, and mean spikes. MED task scores are hard-off. Hard-on survival is an extra column. The protocol uses population 64 and 200 generations. Comparison tables use seed 42; CartPole MED-SNN also reports five seeds $\{0, 1, 2, 3, 42\}$ in Table 5. Acrobot, Pendulum, MountainCar, and all non-CartPole rows remain single-seed (42). Energy quantities are spike proxies. Default ledger cost weights are $c_{\text{spike}} = 1.0$, $c_{\text{syn}} = 0.2$, $c_{\text{leak}} = 0.05$, $c_{\text{exist}} = 0.005$, and $c_{\text{conn}} = 0.0005$, with staged c_{spike} scales $0.3/0.6/1.0$.

4.1 EXPERIMENTAL SETTINGS

Table 1: Environments. Solve thresholds follow the standard Gym cutoffs.

environment	max steps	solve	seed
CartPole-v1	500	steps ≥ 475	42
Acrobot-v1	500	return ≥ -100	42
Pendulum (discrete)	200	return ≥ -300	42
CartPole-1000	1000	steps ≥ 975	42
MountainCar-v0	200	return ≥ -110	42

Hard-off denotes a final test with truncation turned off, used to report task steps and solve. Hard-on keeps the late ledger at test. Death rate is the training fraction of episodes that hit $E \leq 0$. Ranking uses the weighted fitness in equation 1. The staged budget uses $E_{\text{init}} = 300/200/150$ with cuts at generations 40 and 100. Developmental gating is off.

4.2 COMPARISON

Table 2: CartPole-v1, seed 42, population 64, 200 generations. Baselines train on native objectives. MED reports hard-off task scores. EONS-hard uses the same staged hard ledger, a size-penalty fitness, and frozen metabolic genes, following the Schuman et al. genome of thresholds, leak, delays, and weights.

method	score	std	solve	wall (s)	spikes
EVOL	**500.00**	**0.00**	**100%**	1357.7	4281.47
STDP-RL	**500.00**	**0.00**	**100%**	1800.2	4279.39
EONS-proxy	**500.00**	**0.00**	**100%**	1160.7	4279.39
EONS-hard	498.55	6.48	95%	678.8	2232.42
STBP	117.30	13.42	0%	198.5	**365.41**
BPTT-unroll	361.85	94.36	25%	1343.5	1803.74
SpikingJelly RL	109.45	61.27	0%	1268.0	2153.44
e-prop	9.35	0.81	0%	**181.4**	1072.69
CMA-ES	114.85	73.26	0%	869.6	11454.57
MED-SNN	409.85	108.87	45%	886.9	2147.24

On CartPole-v1, seed 42, population 64 and 200 generations, MED-SNN records hard-off mean 409.9 steps (median 463.5, std 108.9) and 45% solve over 20 test episodes. Training reaches 500 steps. Mean training death is 0.77 (early 0.20, mid 0.81, late 0.98), so the ledger is active. Hard-on survival is 0% (mean 23.5 steps under the late ledger): with truncation kept on at test, the champion exhausts the late account before the episode ends. Hard-off scores measure task competence after

Table 3: Acrobot-v1, seed 42, population 64, 200 generations. Baselines train on native objectives. MED reports hard-off task scores.

method	score	std	solve	wall (s)	spikes
EVOL	**-81.65**	12.48	**100%**	1268.7	5071.99
STDP-RL	-95.75	55.07	**100%**	1674.6	5271.98
EONS-proxy	-95.75	55.07	**100%**	1317.7	5271.98
STBP	-500.00	**0.00**	0%	1756.8	11474.99
BPTT-unroll	-500.00	**0.00**	0%	6597.5	20516.42
SpikingJelly RL	-500.00	**0.00**	0%	9885.5	37658.73
e-prop	-500.00	**0.00**	0%	2208.9	36265.11
CMA-ES	-89.10	24.87	80%	867.4	7127.65
MED-SNN	-500.00	**0.00**	0%	**455.6**	981.52

Table 4: Pendulum (discrete torque), seed 42, population 64, 200 generations. Baselines train on native objectives. MED reports hard-off task scores.

method	score	std	solve	wall (s)	spikes
EVOL	-1321.65	272.34	0%	739.7	1137.83
STDP-RL	**-980.97**	145.84	0%	1089.8	1180.33
EONS-proxy	**-980.97**	145.84	0%	814.1	1180.33
STBP	-1534.15	234.67	0%	664.0	5284.74
BPTT-unroll	-1319.90	125.58	0%	2298.8	8997.84
SpikingJelly RL	-1393.80	103.05	0%	3816.0	22258.00
e-prop	-1396.20	**92.87**	0%	821.9	21141.29
CMA-ES	-1202.25	723.38	**25%**	**384.1**	9142.64
MED-SNN	-1423.52	129.22	0%	711.4	456.75

Table 5: CartPole-v1 MED-SNN hard-off over five seeds (population 64, 200 generations). Mean and std are across seeds. Hard-on survival uses the default late ledger.

seed	hard-off steps	solve	train death	hard-on surv.
0	488.15	85%	0.78	0%
1	454.75	70%	0.65	0%
2	472.45	80%	0.64	55%
3	497.80	95%	0.79	0%
42	409.85	45%	0.77	0%
mean±std	464.6 ± 34.7	$75\% \pm 19\%$	0.72 ± 0.08	$11\% \pm 25\%$

search under training-time hard death; they do not claim late-ledger solvency at deployment under default late $R_{\text{surv}} = 2$. The default protocol claim is training-time searchability under hard death: overspending rollouts leave the admissible set during evaluation, while task competence is read out hard-off. Table 9 raises late R_{surv} to 8 and records 100% hard-on survival (mean 494.9 hard-on steps). Extinction reset did not fire. Mean training spikes are 2147. Wall-clock time is 887 seconds. Across five seeds (Table 5), hard-off steps are 464.6 ± 34.7 and solve is $75\% \pm 19\%$, with mean training death 0.72 ± 0.08; default hard-on survival remains low on average ($11\% \pm 25\%$). Literature trainers optimize their native objectives and do not debit the hard ledger during training. Unconstrained EVOL reaches 500 steps and 100% solve with 4281 spikes; EONS-hard under the same staged ledger reaches 498.6 steps and 95% solve. Across five seeds MED-SNN records 464.6 ± 34.7 hard-off steps and $75\% \pm 19\%$ solve with death 0.72 ± 0.08, so the hard-off gap remains visible while search stays open under an active ledger and spikes stay below EVOL. The claim is training-time searchability under hard death, with hard-on solvency as a separate contract (Table 9). EONS-hard freezes metabolic genes and ranks by a size-penalty fitness; MED-SNN co-searches topology, weights, and metabolic coefficients under equation 1. The wider genome enlarges the mixed discrete–continuous space that population 64 over 200 generations must cover, so selection also spends budget on metabolic axes that EONS-hard holds fixed. On this CartPole seed the narrower EONS-hard genome returns higher hard-off solve at lower wall time; MED-SNN remains searchable while exposing that extra dimension. The primary multi-seed claim is CartPole; Acrobot, Pendulum, and MountainCar stay single-seed probes.

On Acrobot-v1 (Table 3), EVOL reaches -81.7 with 100% solve; CMA-ES reaches -89.1 with 80% solve; STDP-RL and EONS-proxy match at -95.8 with 100% solve; STBP, BPTT-unroll,

SpikingJelly RL, and e-prop sit at -500 with 0% solve. MED-SNN records -500, 0% solve, wall 456 seconds, and 982 mean spikes—lowest wall in the row and sparse versus EVOL (5072 spikes). Under the default staged ledger the controllers stay cheap; swing-up does not emerge within 200 generations. Late $E_{\text{init}} = 800$ (Table 12) extends hard-off length to 500 steps with solve still 0%.

On discrete-torque Pendulum (Table 4), every method except CMA-ES records 0% solve against the -300 cutoff. STDP-RL and EONS-proxy share the best mean return (-981). CMA-ES reaches -1202 with 25% solve and wall 384 seconds. MED-SNN records -1424, 0% solve, wall 711 seconds, and 457 mean spikes—the sparsest spike count in the table. On seed 42 only, Acrobot and Pendulum keep MED-SNN searchable under the ledger yet below Gym solve thresholds; these rows are single-seed probes, not five-seed claims. The claim on these rows is training-time search under hard death.

4.3 Ablations

Table 6: Hard truncation versus soft λ penalties on CartPole. Hard-off steps are task steps with truncation off at final test.

variant	hard-off steps	solve	hard-on surv.	train death
hard (default)	409.85	45%	0%	0.77
soft λ=0.1	384.60	45%	0%	0.76
soft λ=1.0	**481.20**	**70%**	0%	0.74
soft λ=3.0	245.95	10%	5%	0.69
soft λ=10.0	298.85	5%	0%	0.75
adaptive Lagrangian	185.10	10%	**10%**	0.75

Table 7: Budget schedules with matched total E_{init} exposure. Hard-off steps are task steps with truncation off at final test. Matched-flat stays tight throughout.

variant	hard-off steps	solve	hard-on surv.	train death
staged (default)	409.85	45%	0%	0.77
linear	366.65	25%	0%	0.49
cosine	397.65	45%	0%	0.57
matched-flat	**441.25**	**60%**	**100%**	0.89

Table 8: Mechanism ablations on CartPole.

variant	hard-off steps	solve	hard-on surv.	train death
staged (default)	409.85	45%	0%	0.77
no hard death	**500.00**	**100%**	0%	0.00
no crossover	451.50	50%	**100%**	0.31
no mutation	24.60	0%	45%	0.72
weight-only	9.40	0%	0%	0.80

Table 9: Hard-on survival on CartPole under the default late $R_{\text{surv}} = 2$ versus a survive-boost variant with late $R_{\text{surv}} = 8$ (mid $R_{\text{surv}} = 6$). Hard-on keeps the late ledger at test.

variant	hard-off steps	solve	hard-on surv.	hard-on steps
late $R_{\text{surv}} = 2$ (default)	409.85	45%	0%	23.5
late $R_{\text{surv}} = 8$	**469.20**	**70%**	**100%**	**494.9**

Under the default staged ledger, search remains open: the champion solves 45% of hard-off tests while late-generation death reaches 0.98. Soft $\lambda = 1.0$ raises hard-off solve to 70% (mean 481.2 steps), but soft penalties reorder fitness without ending trajectories at $E \leq 0$; hard truncation removes overspending rollouts from the admissible set, which soft λ does not do. No hard death reaches 500 steps and 100% solve with training death 0, confirming that the hard boundary is what keeps death active under the default contract. Linear and cosine stay searchable (25% and 45% solve). Matched-flat records 441.3 hard-off steps, 60% solve, and 100% hard-on survival. The schedule changes only the feasibility boundary of evaluation across generations; the task distribution is unchanged, so the contrast with curriculum learning is that curriculum varies task difficulty while the budget schedule varies ledger strictness. This paper does not train a separate curriculum-learning baseline that rewrites the Gym task distribution; Table 7 is the experimental contrast under a fixed task. A future matched curriculum study that varies task difficulty while holding the ledger

fixed would complement that table. Removing mutation or restricting search to weights drops hard-off steps to 24.6 and 9.4. Default hard-on survival is 0% (mean 23.5 steps) under late $R_{\text{surv}} = 2$; Table 9 records the survive-boost contract with late $R_{\text{surv}} = 8$, where hard-on survival becomes 100% (mean 494.9 steps). The default column therefore bounds searchability under the tight late ledger; hard-on solvency is reported only when the survival allowance is raised as in that table.

4.4 E-PROP UNDER THE SAME HARD-DEATH PROTOCOL

Unconstrained e-prop sits in the comparison table. The same discrete-action approximation with the hard ledger is reported in Table 12.

4.5 SENSITIVITY, ROBUSTNESS, AND BOUNDARY PROBES

Table 10: Late-ledger neighbourhood and isometric E_{init} rescaling on CartPole. Late R_{task} is varied with early and mid knobs held fixed.

variant	hard-off steps	solve	hard-on surv.	train death
default late 150/2/1.0	409.85	45%	0%	0.77
late E_init=120	409.50	35%	0%	0.77
late E_init=180	409.20	35%	0%	0.75
late R_surv=1	405.10	30%	0%	0.78
late R_surv=4	**500.00**	**100%**	25%	0.72
late R_task=1	399.60	35%	0%	0.78
late R_task=4	452.50	70%	**90%**	0.75
late c_spike=0.7	424.50	40%	0%	0.75
late c_spike=1.3	405.30	35%	0%	0.78
w_energy=0.0003	466.80	80%	0%	0.68
w_energy=0.003	477.75	70%	5%	0.78
iso-scale 0.8	203.25	20%	0%	0.79
iso-scale 1.2	428.30	40%	0%	0.74

Table 11: Robustness: observation noise, initial-state scale, width, and a 1000-step horizon.

variant	hard-off steps	solve	hard-on surv.	train death
default 64 neurons / 500 steps	409.85	45%	0%	0.77
obs. noise 0.05	469.05	**80%**	0%	0.70
obs. noise 0.10	415.50	55%	0%	0.83
init scale 1.5	469.80	75%	0%	0.72
32 neurons	383.10	25%	0%	0.80
96 neurons	454.70	65%	0%	0.74
1000 steps	**669.25**	50%	0%	0.73

Table 12: Boundary probes: MountainCar, Acrobot late $E_{\text{init}} = 800$, and hard-death e-prop. The e-prop hard-on column is 0% because training death is 1.00 and the champion lasts 7 hard-off steps.

variant	hard-off steps	solve	hard-on surv.	train death
MountainCar MED	200.00	0%	0%	0.45
MountainCar EVOL	200.00	0%	0%	0.00
Acrobot late E_init=800	**500.00**	0%	0%	0.98
e-prop hard death	7.00	0%	0%	1.00

Table 13: CMA-ES and BPTT-unroll on CartPole with a thin bound ledger ($E_{\text{init}} = 150$, $R = 2$, $c_{\text{spike}} = 4.5$), seed 42, 200 generations. Test scores are hard-off. Training death is the fraction of rollouts that hit $E \leq 0$.

method	hard-off steps	solve	train death	wall (s)
CMA-ES tight	**90.55**	0%	0.95	104.3
BPTT-unroll tight	9.55	0%	1.00	**23.9**

On the completed CartPole main run, extinction reset counted 0. Late $R_{\text{task}} = 1$ stays searchable (399.6 steps, 35% solve). Late $R_{\text{task}} = 4$ records 452.5 steps and 70% solve. MountainCar solve is 0% for MED and EVOL. Acrobot with late $E_{\text{init}} = 800$ lasts 500 hard-off steps with 0% solve. Hard-death e-prop records training death 1.00 and 7.0 hard-off steps. Under the thin bound ledger

(Table 13), CMA-ES training death is 0.95 with hard-off 90.6 steps and 0% solve; BPTT-unroll training death is 1.00 with hard-off 9.6 steps and 0% solve. Seed 42 never records a 50% solve on this thin ledger: the solve column is 0% for both rows, and the 90.6 hard-off steps are a short truncated mean, not a Gym solve. Short hard-off length under a thin ledger is failure to keep long trajectories; MED-SNN under the staged hard ledger remains searchable on CartPole while these fixed-topology local trainers do not. Late c_{spike} scales 0.7 and 1.3 in Table 10 probe spike-cost sensitivity; c_{syn}, c_{leak}, c_{exist}, and c_{conn} stay at the defaults above and are not swept independently in this draft.

5 Conclusion

MED-SNN trains gene-encoded spiking controllers under a hard energy ledger. Two mechanisms organize the search. Hard truncation ends a trajectory when the account reaches the death threshold, so overspending rollouts leave the admissible set. Inter-generational budget matching starts loose and tightens toward the target ledger, so generation 0 still produces a selection signal. Ranking uses the weighted fitness in equation 1. On CartPole-v1 the pipeline remains searchable under training-time hard death (death rate 0.77), with hard-off 409.9 ± 108.9 steps and 45% solve. Hard-off task scores and hard-on survival are reported separately. On the matched schedule table, matched-flat records 441.3 hard-off steps, 60% solve, and 100% hard-on survival. Staged, linear, and cosine solve at 45%, 25%, and 45%. A discrete-action e-prop probe is evaluated under the same hard-death protocol. On Acrobot, aligning late E_{init} with the swing-up horizon is a boundary probe.

Limitations. The main evidence is CartPole. MountainCar remains a negative result. Energy is a spike proxy. Default late-ledger hard-on survival can sit near zero; that column is reported separately from hard-off task scores. The e-prop comparison is a discrete-action approximation. Future work measures inference cost on neuromorphic hardware, adapts the budget from population survival, and maps further local learning rules under the same truncation protocol.

AI use statement

(This section is **required** and does not count toward the page limit.)

We used generative AI tools to assist with English drafting, LaTeX formatting, and experiment summaries. The authors reviewed and take responsibility for all claims, numbers, and final wording.

See the ICLR 2027 AI Policy for Authors for more details.

Ethics statement

(This section is **recommended** and does not count toward the page limit.)

This work uses simulated Gym-style control tasks only. It involves no human subjects, personal data, or physical robot trials.

Reproducibility statement

(This section is **recommended** and does not count toward the page limit.)

Code, paper-aligned YAML configs, and run scripts are in the accompanying repository. CartPole multi-seed and ablation metrics regenerate from those configs with the reported seeds, population, and generation count.

References

Guillaume Bellec, Franz Scherr, Anand Subramoney, Elias Hajek, Darjan Salaj, Robert Legenstein, and Wolfgang Maass. A solution to the learning dilemma for recurrent networks of spiking neurons. *Nature Communications*, 11:3625, 2020. doi: 10.1038/s41467-020-17236-y.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 41–48, 2009. doi: 10.1145/1553374.1553380.

Wei Fang, Yanqi Chen, Jianhao Ding, Zhaofei Yu, Timothée Masquelier, Ding Chen, Liwei Huang, Huihui Zhou, Guoqi Li, and Yonghong Tian. SpikingJelly: An open-source machine learning infrastructure platform for spike-based intelligence. *Science Advances*, 9(40):eadi1480, 2023. doi: 10.1126/sciadv.adi1480.

Tanguy Fardet and Anna Levina. Simple models including energy and spike constraints reproduce complex activity patterns and metabolic disruptions. *PLOS Computational Biology*, 16(12): e1008503, 2020. doi: 10.1371/journal.pcbi.1008503.

Andrea Ferigo, Elia Cunegatti, and Giovanni Iacca. Neuron-centric Hebbian learning. In *Proceedings of the Genetic and Evolutionary Computation Conference*, 2024. doi: 10.1145/3638529.3654011.

Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. doi: 10.1162/106365601750190398.

Daniel Hasegan, Matt Deible, Christopher Earl, David D'Onofrio, Hananel Hazan, Haroon Anwar, and Samuel A. Neymotin. Training spiking neuronal networks to perform motor control using reinforcement and evolutionary learning. *Frontiers in Computational Neuroscience*, 16:1017284, 2022. doi: 10.3389/fncom.2022.1017284.

Ismael Jaras, Marcos E. Orchard, Pedro E. Maldonado, and Rodrigo C. Vergara. Unveiling the role of local metabolic constraints on the structure and activity of spiking neural networks. *PLOS Computational Biology*, 21(6):e1013148, 2025. doi: 10.1371/journal.pcbi.1013148.

Sen Lu and Abhronil Sengupta. Neuroevolution guided hybrid spiking neural network training. *Frontiers in Neuroscience*, 16:838523, 2022. doi: 10.3389/fnins.2022.838523.

Elias Najarro, Shyam Sudhakaran, and Sebastian Risi. Towards self-assembling artificial neural networks through neural developmental programs. In *Proceedings of the 2023 Artificial Life Conference*. MIT Press, 2023. doi: 10.1162/isal_a_00697.

Emre O. Neftci, Hesham Mostafa, and Friedemann Zenke. Surrogate gradient learning in spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019. doi: 10.1109/MSP.2019.2931595.

Wenxuan Pan, Feifei Zhao, Bing Han, Yiting Dong, and Yi Zeng. Emergence of brain-inspired small-world spiking neural network through neuroevolution. *iScience*, 27(2):108845, 2024. doi: 10.1016/j.isci.2024.108845.

Sidney Pontes-Filho, Kristoffer Olsen, Anis Yazidi, Michael A. Riegler, Pål Halvorsen, and Stefano Nichele. Towards the neuroevolution of low-level artificial general intelligence. *Frontiers in Robotics and AI*, 9:1007547, 2022. doi: 10.3389/frobt.2022.1007547.

Catherine Rodriquez and James Ghawaly. Co-evolved spiking neural network ensembles via marginal contribution fitness. In *Proceedings of the 2026 ACM International Conference on Neuromorphic Systems*, 2026. arXiv:2606.13985.

Catherine D. Schuman, J. Parker Mitchell, Robert M. Patton, Thomas E. Potok, and James S. Plank. Evolutionary optimization for neuromorphic systems. In *Proceedings of the Neuro-Inspired Computational Elements Workshop*, pp. 1–9, 2020. doi: 10.1145/3381755.3381758.

Kenneth O. Stanley and Risto Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2):99–127, 2002. doi: 10.1162/106365602320169811.

Yujie Wu, Lei Deng, Guoqi Li, Jun Zhu, and Luping Shi. Spatio-temporal backpropagation for training high-performance spiking neural networks. *Frontiers in Neuroscience*, 12:331, 2018. doi: 10.3389/fnins.2018.00331.

Larry Yaeger. Computational genetics, physiology, metabolism, neural systems, learning, vision, and behavior or PolyWorld: Life in a new context. In Christopher G. Langton (ed.), *Artificial Life III*, pp. 263–298. Addison-Wesley, 1994.