One Spike Decision Reinforcement Learning Framework for Dynamic Environments

Anonymous authors
Paper under double-blind review

Abstract

Deep reinforcement learning (DRL) agents face challenges in natural environments that are similar to those encountered by biological organisms: they must make actions that are both accurate and timely in response to dynamic, non-stationary conditions. However, achieving such behavior incurs significant computational overhead, limiting the scalability of DRL in real-world applications. Spiking Neural Networks (SNNs), as the most biologically plausible computational model of neurons, offer a promising energy-efficient alternative for reinforcement learning due to their low computational cost. Existing SNN-based methods, however, often rely on multiple simulation time steps to approximate analog activations, which compromises their low-latency and low-power advantages. To address this, we propose a novel DRL framework based on one-spike firing decision (OSFD), which redefines the use of SNNs in DRL. In OSFD, each decision step triggers only a single spike to produce an action, while the residual membrane potential is incrementally accumulated across steps. In addition, we introduce Bayesian variational inference to dynamically regulate the contribution of residual potentials based on state information gain, thereby optimizing policy learning. Experimental results demonstrate that our method not only surpasses conventional artificial neural network (ANN)-based frameworks in performance but also significantly reduces computational cost.

1 Introduction

Deep reinforcement learning (DRL) algorithms have demonstrated exceptional potential in complex decision-making tasks, such as robotic control Li et al. (2024b); Gu et al. (2024) and autonomous driving Chowdhury et al. (2024); Karnchanachari et al. (2024). In real-world scenarios characterized by limited observability and non-stationary environmental dynamics, the ability of an agent to make high-frequency and precise decisions becomes crucial. However, achieving agile motion planning and optimal performance incurs significant computational costs, making large-scale deployment in real-world applications impractical Dong et al. (2017).

Natural intelligence systems offer an inspiring solution to the aforementioned dilemma: biological organisms exhibit an impressive balance between extremely low-cost decision-making and highly agile responses when faced with similar challenges. As the closest computational model to biological neuronal firing, Spiking Neural Networks (SNNs) are theoretically well-suited to break the energy-efficiency bottlenecks of conventional DRL systems Deng et al. (2024); Ochs et al. (2024); Shi et al. (2024). However, existing SNN-based reinforcement learning methods largely adopt the operational paradigm of artificial neural networks (ANNs) Jiang et al. (2024); Ding et al. (2021); Li & Zeng (2022). In particular, mainstream directly encoded SNNs extend the temporal simulation window length T to approximate the activation precision of ANNs. This approach overlooks the intrinsic structural characteristics and potential advantages of SNNs, failing to fully exploit their dynamic properties and temporal modeling capabilities. Moreover, repeated computations along the time dimension can compromise the low-cost and low-latency benefits that distinguish SNNs Shen et al. (2024a).

This leads us to consider whether an SNN could perform the complete input-to-action mapping within a single simulation time window, thereby substantially reducing computational cost. However, such a strategy

inevitably results in a significant drop in computational precision Han et al. (2020); Hao et al. (2023b); Hu et al. (2023). Building upon this idea, our study further discovers that SNNs inherently possess temporal dynamic processing capabilities Qiao et al. (2024); Rathi & Roy (2024); Cao et al. (2024). Specifically, their spike-driven membrane potential accumulation mechanism offers a natural form of temporal consistency in decision-making, suggesting promising potential for precision enhancement. This indicates a fundamental alignment between the biological characteristics of SNNs and the demands of reinforcement learning: the event-driven nature of spike-based computation and the inter-step integration of membrane dynamics together form a synergistic mechanism for decision response and continual optimization. Therefore, we argue that SNNs should be redefined as a novel computational paradigm grounded in their inherent temporal properties, in order to fully unleash their potential in reinforcement learning.

In this work, we propose a One-Spike Firing Decision (OSFD) framework to reconstruct the Deep Reinforcement Learning paradigm for SNN. Instead of relying on the multi-timestep averaging computation typical of traditional SNNs, this framework directly outputs actions at each step. Concurrently, it achieves incremental policy optimization by constraining the accumulation of membrane potentials across successive decisions. Furthermore, to fully capture dynamic dependencies across timesteps, we introduce Bayesian Variational Inference to quantify the state information gain induced by actions. This gain is then utilized as intrinsic feedback to enable the adaptive, dynamic adjustment of the membrane potential leakage factor. Experimental results across multiple environments Li et al. (2024a); Walraven et al. (2025), based on a series of classic DRL algorithms Mnih et al. (2013), demonstrate that OSFD facilitates fine-grained, precise decision-making with extremely low energy consumption while attaining higher cumulative rewards.

The main contributions of this work are summarized as follows:

1. This paper proposes a reinforcement learning framework based on one-spike firing decision (OSFD), which breaks away from traditional SNN approaches that rely on multi-step spike averaging. OSFD enables one-spike action execution at each decision step, effectively overcoming precision loss while significantly reducing computational cost.

2. We theoretically model the state information gain resulting from actions through the lens of Bayesian Variational Inference, leading to the proposal of a lightweight adaptive membrane potential constraint mechanism. This facilitates a theoretically grounded incremental policy optimization approach for SNN.

3. We observe that spiking neural networks are inherently well-suited for reinforcement learning. Experimental results confirm that our method can surpass conventional artificial neural network frameworks in RL tasks, offering new insights for future research directions.

2 Related Work

2.1 Reinforcement Learning with Spiking Neural Networks

Several studies have explored the integration of spiking neural networks with reinforcement learning. For example, PopSAN Tang et al. (2021) introduced a population-coded spiking actor network to enable energy-efficient transfer learning for robotic control. Compared to mainstream DRL algorithms, this approach achieves comparable performance while significantly reducing inference energy consumption. Luca et al. Zanatta et al. (2024) trained a novel SNN framework using the PPO algorithm, demonstrating its effectiveness in relevant tasks and achieving faster training speeds. Oikonomou et al. Oikonomou et al. (2023) designed a hybrid SNN-DDPG algorithm to leverage the low-power advantages of SNNs in robotic tasks. Gui et al. Liu et al. (2022) proposed DSQN, which directly trains a spiking deep Q-network within a deep spiking reinforcement learning framework, achieving strong performance in Atari games. Chen et al. Chen et al. (2022) utilized the membrane voltage of non-spiking neurons to represent Q-values, resulting in lower energy consumption and enhanced robustness. ILC-SAN Chen et al. (2024) introduced multiple population neurons to decode actions across different dimensions, establishing temporal and spatial connections among these neurons to form a spiking actor network with performance comparable to mainstream DRL methods.

2.2 Timestep Reduction under SNN Rate Coding

Many researchers have proposed methods to reduce the number of timesteps in spiking neural networks (SNNs). DT-SNN Li et al. (2023b) dynamically determines the number of timesteps during inference by computing the cumulative output entropy at each timestep, adapting to input-dependent variations. SEENN Li et al. (2023a) treats the number of timesteps as a variable, allowing the network to exit early based on different inputs. Chowdhury et al. Chowdhury et al. (2022) introduced the concept of pruning to optimize SNNs for both visual and sequential tasks. Datta et al. Datta et al. (2023) proposed a training framework that dynamically allocates timesteps for each Vision Transformer (ViT) module. Xiao et al. Song et al. (2024) developed OST, which integrates temporal compression with compensation components, significantly reducing the spatiotemporal overhead of spiking Transformers.Furthermore, SSNN Ding et al. (2024) introduced a contraction-based neural network that progressively reduces both timesteps and redundancy, while TIM Shen et al. (2024b) proposed a temporal interaction module that effectively leverages missing temporal information, further enhancing efficiency.

3 Preliminary

3.1 Spiking Neural Model

In an artificial neural network, the input $\boldsymbol{a}^{l-1}$ to layer l will be transformed by a linear transformation matrix and a nonlinear activation function to obtain a mapping output $\boldsymbol{a}^l$. Formally, for any layer $l = 1, 2, 3, \ldots, L$ in the network, we have: $\boldsymbol{a}^l = g\left(\boldsymbol{W}^l \boldsymbol{a}^{l-1}\right)$ where $g(\cdot)$ usually uses ReLU as the activation function. Unlike ANNs, SNN uses binary spikes to transmit information. This paper uses the well-established Leaky-Integrate-and-Fire (LIF) Huang et al. (2024) neuron model to introduce the unique spatiotemporal dynamic characteristics of the spike model. In the LIF neuron model, the dynamic process of its membrane potential update is as follows:

$$\boldsymbol{v}^l(t) = \tau \boldsymbol{v}^l(t-1) + \boldsymbol{W}^l \theta^{l-1} \boldsymbol{s}^{l-1}(t) - \theta^l \boldsymbol{s}^l(t) \tag{1}$$

Where $\boldsymbol{v}^l(t)$ denotes the membrane potential of neurons in layer l at time step t, $\boldsymbol{W}^l$ denotes the corresponding linear transformation matrix, θ^l represents the spike firing threshold, $\boldsymbol{s}^{l-1}$ is the binary output spike of the neuron in the previous layer, and $\boldsymbol{s}^l$ is defined as follows:

$$\boldsymbol{s}^l(t) = H\left(\boldsymbol{u}^l(t) - \theta^l\right) = \begin{cases} 1 & \text{if } \boldsymbol{u}^l(t) > \theta^l, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

Where $H(\cdot)$ represents the Heaviside step function, which means that when the membrane potential $\boldsymbol{u}^l(t)$ exceeds the threshold θ^l , the neuron will generate an output spike. In this process, the membrane potential will be reset by subtracting the threshold or directly changing to 0. Before the neuron emits a spike, $\boldsymbol{u}^l(t) = \tau \boldsymbol{v}^l(t-1) + \boldsymbol{W}^l \theta^{l-1} \boldsymbol{s}^{l-1}(t)$ denotes the temporary membrane potential of the upcoming spike.

3.2 SNN and Quantized ANNs

The activation strength of an SNN neuron is represented by its average firing rate over T time steps, i.e., $s = \frac{1}{T} \sum_{t=1}^{T} s(t)$, Assuming a consistent total activation $\sum_t q_s(t) = q_a = a$, the activation of an SNN can be approximated by an ANN with T discrete levels, formulated as:

$$s \cdot T = a - v(T) = \theta \cdot \operatorname{clip}\left(\left\lfloor \frac{a}{\theta} \right\rfloor, 0, T\right), \tag{3}$$

In this formula, θ is the quantization gap, $\operatorname{clip}(x, 0, T)$ restricts the value within the interval $[0, T]$, $v(T)$ denotes the residual potential, representing the error introduced by quantization. Accordingly, the firing rate in SNNs can be viewed as a quantized approximation of ANN activations, where the approximation error arises from the discretization gap θ and the residual potential. As T increases, finer quantization becomes possible. To control quantization error, θ is typically set as $\theta \approx \frac{\max(a)}{T}$, in order to ensure a finer approximation of the original activation values.

4 One Spike Decision Framework

In this section, we first construct an equivalent model between artificial neural networks and spiking neural networks to highlight the differences among various decision-making strategies. We then present a detailed description of the proposed One-Spike Firing Decision reinforcement learning framework. Furthermore, we theoretically derive the necessary condition that the optimal membrane potential distribution should satisfy, and consequently constrain the residual potential accumulation mechanism based on state information gain.

4.1 Strategy Comparison Based on Equivalent Modeling

To better understand the architecture of our proposed method, it is necessary to establish an equivalence model between ANNs and SNNs. Figure 1 illustrates different DRL sampling paradigms, including our proposed strategy.

While SNNs offer low-power computation, this benefit often comes at the cost of reduced numerical precision. To compensate, SNNs typically extend the simulation time window to approximate the activation accuracy of ANNs. As discussed theoretically in Section 3.2, the spike firing rate in SNNs can be viewed as a quantized approximation of ANN activations. Accordingly, in the context of DRL decision-making, a vanilla ANN (Figure 1 (a)) can be interpreted as a binary sequence-like SNN (Figure 1 (b)) with $T = n$ time steps. In this setup, S_t^i denotes the input at time step t during the i-th iteration, and O_t^i represents the corresponding output. In the vanilla SNN, inputs across different time steps are repeated samplings of the original input, such that $S_t^i = S_{t+1}^i$.

However, such a computational paradigm becomes suboptimal in the context of DRL. In complex and dynamic environments, repeated computations across multiple time steps introduce latency and additional overhead, which contradicts the inherent low-cost advantage of SNNs, as illustrated in (b). To address this, we align the decision-making process with DRL's per-step structure by employing a one-timestep decision mechanism. This corresponds to the equivalent model shown in (c), where an activation-quantized ANN can be interpreted as a $T = 1$ SNN without potential accumulation. Moreover, as shown in (d), our method leverages the temporal features embedded in the residual potential from previous steps, allowing it to be continuously accumulated and utilized to gain additional informational advantages. This intrinsic characteristic of spiking neural networks not only significantly reduces computational cost but also enhances the network's capacity through the presence of residual potential.

4.2 OSFD Framework embedded into DRL Algorithms

In this section, we present the implementation details of OSFD and explain its sampling and training procedures. The proposed framework can be integrated into standard DRL algorithms during training. As illustrated in Figure 2, during the sampling phase, the network predicts the action-value function $Q(s, a)$ given an observation s, and generates an action a that transitions the agent to the next state s_{t+1}. We replace the traditional ReLU activation function with Leaky Integrate-and-Fire (LIF) neurons. LIF neurons transmit information through spike signals and the accumulation of membrane potential, emitting a spike to perform decision-making.

After each single-step sampling, the residual membrane potential from time step $t-1$ is stored in the experience replay buffer along with the corresponding state, action, and other relevant information. Within an episode, the residual potentials at different time steps are continuously accumulated and not reset. Therefore, we aggregate the remaining potential from the previous decision step to constrain the computation at the next time step. At time step t, the input to layer l of the network is given by (v_t^l, s_t), where s_t denotes the current observation and v_t^l represents the membrane potential of the l-th layer LIF neuron at time t, which is the residual potential remaining after spike emission at time $t-1$. The membrane potential dynamics for layer l at time t are expressed as: $\boldsymbol{v}_t^l = \tau \boldsymbol{v}_{t-1}^l + \boldsymbol{W}^l \theta^{l-1} \boldsymbol{s}_t^{l-1} - \theta^l \boldsymbol{s}_t^l$

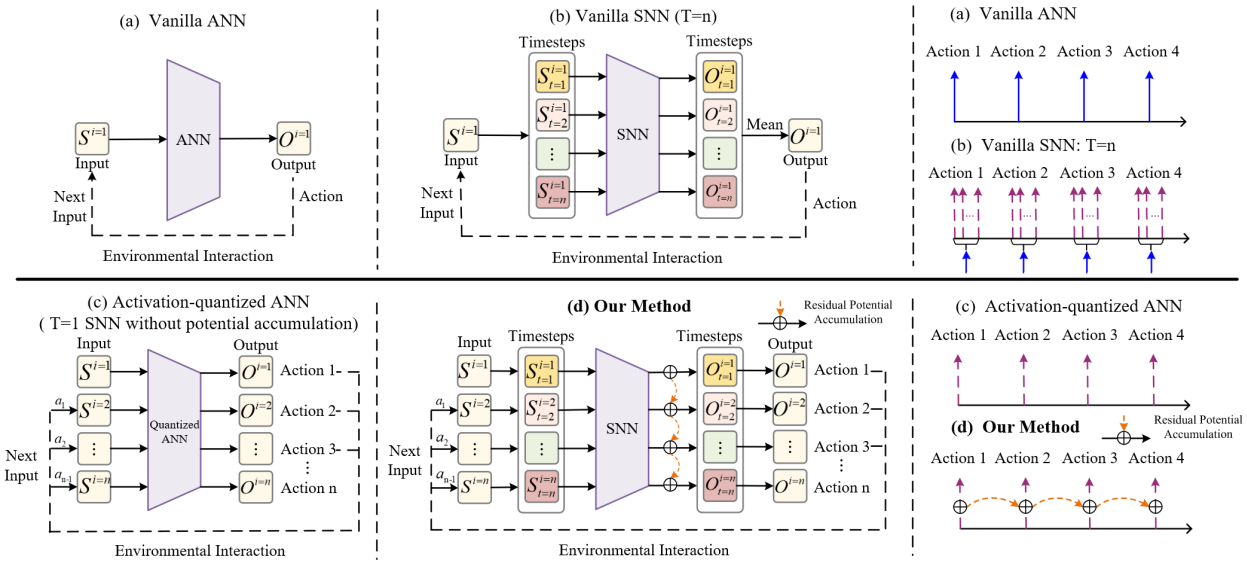

Figure 1: Comparison between our proposed (d) one-spike reinforcement learning paradigm and (a) vanilla ANN, (b) vanilla SNN (with $T = n$), and (c) activation-quantized ANN. In (b), multiple simulation time steps are used to approximate the computational precision of ANNs, making (a) and (b) functionally equivalent. Both (a) and (b) incur higher latency and computational cost per decision compared to (c) and (d), but provide greater computational accuracy. The model in (c) can also be interpreted as a single-step SNN with $T = 1$, which differs from (d) in that it lacks residual potential accumulation.

5 Potential Accumulation Mechanism Based on Dynamic Information Gain

In complex and non-stationary dynamic environments, reinforcement learning agents must strike a crucial balance between decision speed and accuracy. Spiking Neural Networks utilizing one-spike decision-making offer a lightweight and low-latency computational paradigm. This approach circumvents the limitations of traditional SNNs, which rely on multi-timestep simulation approximations, and compensates for the loss of temporal information by accumulating residual membrane potentials across steps.

However, this mechanism encounters a critical challenge in practical applications: the unselective accumulation of membrane potentials may introduce historical noise, leading to erroneous state estimations by the agent. To address this issue, this section proposes a dynamical Potential Adjustment and Decision-making mechanism. Grounded in rigorous mathematical derivation, this mechanism quantifies the information gain of actions regarding future states, utilizing this to dynamically constrain the membrane potential accumulation process and achieve highly efficient, incremental policy optimization.

5.1 Mutual Information-Driven Variational Posterior Approximation

In this section, we formally model the problem. In reinforcement learning, a high-quality state representation should capture information intricately tied to environmental dynamics, thereby facilitating effective function approximation and policy optimization. To this end, we introduce a unified metric for state information gain. First, through Bayesian variational inference, we derive the optimal distribution conditions for the residual membrane potential and construct a theoretically grounded dynamic constraint for membrane potential accumulation. Subsequently, we characterize the contraction effect of this information gain on the Bellman residual, theoretically demonstrating the effectiveness of the proposed method. We begin by presenting the following lemma, which is grounded in mutual information-driven exploration theory Houthooft et al. (2016); Hao et al. (2023a).

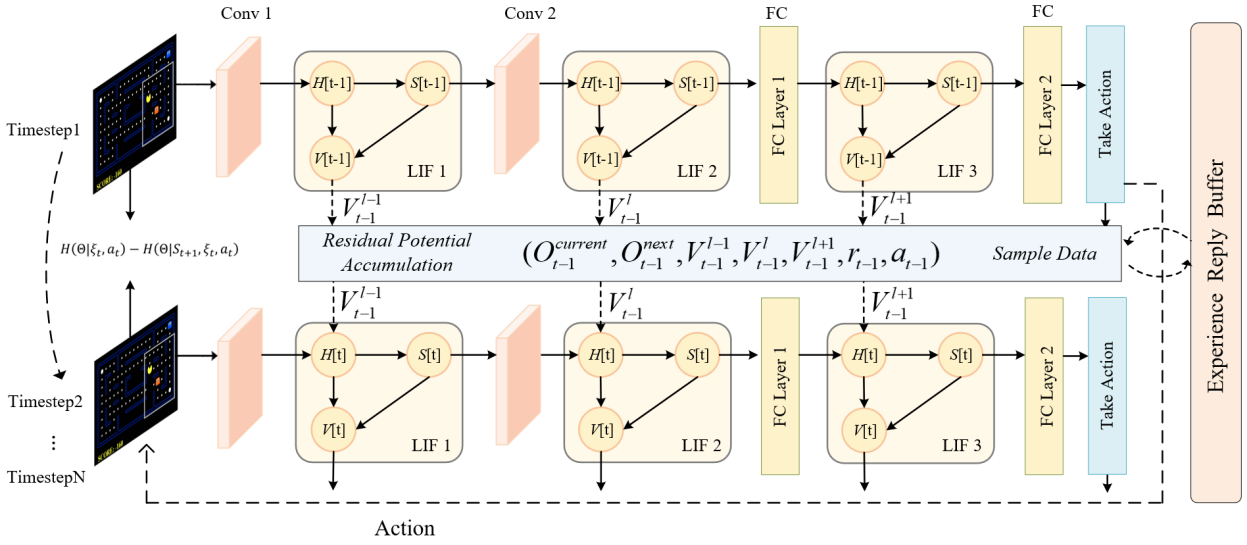

Figure 2: OSFD employs a spiking neural network to generate the next-step action within a single simulation window, while dynamically accumulating residual membrane potentials. The accumulation process is further regulated based on the state information gain at each step.

Lemma 5.1 (Mutual Information-Driven Policy Incentive)**.** *In this setting, we maintain a model of the environmental dynamics $p(s_{t+1}|s_t, a_t, \theta)$ and treat the unknown model parameters as a random variable Θ, with its specific realization being $\theta \in \Theta$. Given the agent's history up to time step t, denoted as $\xi_t = \{s_1, a_1, \ldots, s_t\}$, and the current action a_t, we define the information gain for exploration as the conditional mutual information between the next state S_{t+1} and the model parameters Θ. According to information theory:*

$$I(S_{t+1}; \Theta|\xi_t, a_t) = \mathbb{E}_{s_{t+1},\theta\sim p(s_{t+1},\theta|\xi_t,a_t)}\left[\log \frac{p(s_{t+1}, \theta|\xi_t, a_t)}{p(s_{t+1}|\xi_t, a_t)p(\theta|\xi_t, a_t)}\right] \tag{4}$$

This mutual information can be equivalently expanded as the difference in conditional entropy, or the expected KL divergence between the posterior and the prior:

$$\begin{aligned} I(S_{t+1}; \Theta|\xi_t, a_t) &= H(\Theta|\xi_t, a_t) - H(\Theta|S_{t+1}, \xi_t, a_t) \\ &= \mathbb{E}_{s_{t+1}\sim p(s_{t+1}|\xi_t,a_t)}\Big[D_{\mathrm{KL}}[p(\theta|\xi_t, a_t, s_{t+1}) \,\|\, p(\theta|\xi_t, a_t)]\Big] \end{aligned} \tag{5}$$

Here, $p(\theta|\xi_t, a_t)$ represents the prior belief distribution of the parameters before executing action a_t at time step t, while $p(\theta|\xi_t, a_t, s_{t+1})$ denotes the updated posterior distribution after observing the new state s_{t+1}. The aforementioned mutual information can be further expressed as the expected KL divergence between the posterior and prior distributions. Specifically, the KL divergence term within the brackets of the above equation defines the Bayesian Surprise Itti & Baldi (2005) given a specific state transition.

Remark: In this lemma, the mutual information $I(S_{t+1}; \Theta|\xi_t, a_t)$ measures the expected amount of information that observing the next state S_{t+1} provides for reducing the uncertainty of the model parameters Θ, conditioned on the history ξ_t and the current action a_t.

Lemma 5.2 (Variational Bayesian Posterior Approximation and Information Gain Metric)**.** *Let $q(\theta;\, \phi)$ be the variational distribution used to approximate the intractable posterior $p(\theta \mid \mathcal{D})$, and assume it is a fully factorized Gaussian distribution,$\mathcal{D} = \{\xi_t, a_t, s_{t+1}\}$ to represent the current observation:*

$$q(\theta;\, \phi) = \prod_{i=1}^{|\theta|} \mathcal{N}(\theta_i \mid \mu_i, \sigma_i^2), \qquad \phi = \{\mu, \sigma\}. \tag{6}$$

Minimizing the KL divergence between $q(\theta;\,\phi)$ and the true posterior is equivalent to maximizing the Evidence Lower Bound (ELBO), whose objective function can be expressed as:

$$\mathcal{L}[q(\theta;\,\phi),\mathcal{D}] = \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D}\mid\theta)] - D_{\mathrm{KL}}[q(\theta;\phi)\,\|\,p(\theta)]. \tag{7}$$

During the policy update process, for any given parameter update step $\Delta\phi$, the KL divergence between the variational distributions before and after the update—which represents the induced intrinsic information gain—can be approximated via a second-order Taylor expansion using the Fisher information matrix:

$$D_{\mathrm{KL}}[q(\theta;\,\phi_t+\Delta\phi)\,\|\,q(\theta;\,\phi_t)] \approx \frac{1}{2}\Delta\phi^\top H(\phi_t)\Delta\phi, \tag{8}$$

where $H(\phi_t) = \nabla^2_\phi D_{\mathrm{KL}}[q(\theta;\phi)\,\|\,q(\theta;\phi_t)]\big|_{\phi=\phi_t}$ is the Hessian matrix under the current parameters.

Remark: This lemma indicates that regardless of the specific parameter update rule $\Delta\phi$ adopted by the network, the resulting distribution change, i.e., the information gain, can be measured by the quadratic form of the parameter update on the Riemannian manifold defined by the Hessian matrix H. This provides a mathematical foundation for the subsequent quantification of the additional contribution that the SNN residual membrane potential mechanism provides to policy exploration.

5.2 Theoretical Modeling of Potential Information Gain

In the one spike decision-making framework, SNN neurons possess a unique temporal memory property: the residual membrane potential decays and accumulates across time steps, subsequently affecting future information transmission. Therefore, this study introduces an additional historical memory term into the variational update rule of the policy parameters, aiming to reflect the unique computational characteristics of the SNN membrane potential mechanism at the optimization level.

Definition 5.3 (Effect of Residual Membrane Potential on Policy Parameter Updates)**.** In our proposed method, the parameter updates of the SNN are driven not only by the gradient of the current data but are also modulated by the accumulative effect of the historical potential. The update rule is defined as:

$$\phi_{t+1}^{\mathrm{SNN}} = \phi_t + \nabla_\phi \mathcal{L}_{\mathrm{data}}^{(t)} + \lambda \nabla_\phi \mathcal{L}_{\mathrm{mem}}^{(t)}, \tag{9}$$

where $\nabla_\phi \mathcal{L}_{\mathrm{data}}^{(t)} = \nabla_\phi \mathcal{L}_{\mathrm{data}}(\phi_t; s_t, a_t, s_{t+1})$ denotes the direct gradient generated by the current state transition; $\nabla_\phi \mathcal{L}_{\mathrm{mem}}^{(t)} = \nabla_\phi \mathcal{L}_{\mathrm{mem}}(\phi_t; V_{t-1})$ corresponds to the residual memory gradient introduced by the historical membrane potential V_{t-1}; and $\lambda > 0$ is the weighting control factor for the memory gradient.

To explore the conditions for the information gain of this method, we first formally define the temporal properties of the gradients within the reinforcement learning trajectory. In continuous Markov Decision Processes , both the environmental dynamics and the policy function typically exhibit smoothness, making the observation data from adjacent time steps highly correlated. Herein, we introduce the following assumption:

Assumption 5.4 (Temporal Gradient Correlation)**.** Assume that the environmental dynamics model and the policy network are sufficiently smooth within the parameter space, and that the trajectory sequences sampled by the agent possess temporal continuity. For a given time step t and any past time step $t-k$ ($k \geq 1$), their corresponding data log-likelihood gradients satisfy an exponentially decaying positive correlation on the Riemannian manifold defined by the Hessian matrix H:

$$\mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\mathrm{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\mathrm{data}}^{(t-k)}\right)\right] \geq \rho^k C, \tag{10}$$

where $C = \mathbb{E}\big[\|\nabla_\phi \mathcal{L}_{\mathrm{data}}\|_H^2\big] > 0$ is the expected squared Riemannian norm of the single-step gradient, and $\rho \in (0,1)$ is the temporal correlation coefficient of the MDP trajectory, which depends on the smoothness of the environment and the sampling frequency.

Based on the above assumption, combined with the leaky integrate-and-fire mechanism unique to SNNs, we can rigorously derive the geometric relationship between the memory gradient and the current data gradient:

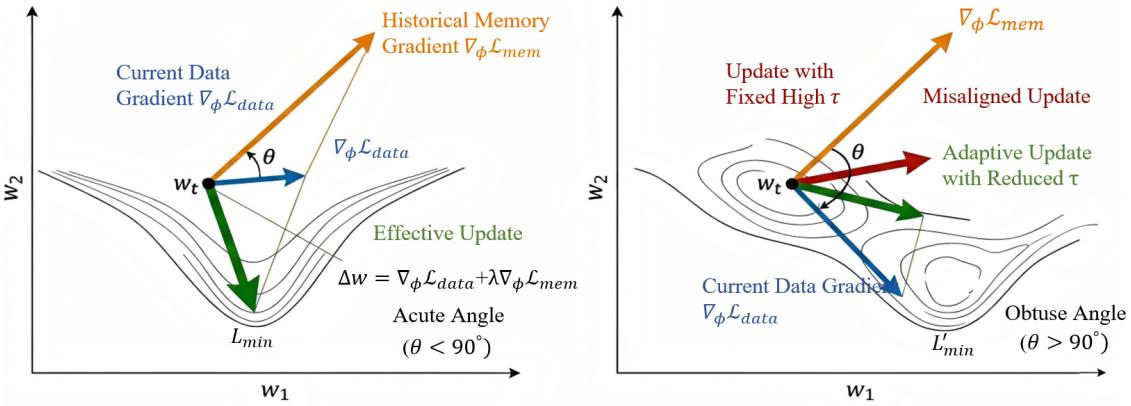

Figure 3: Geometric illustration of gradient interference in SNNs. **(Left) Stationary environment:** The data gradient $\nabla_\phi \mathcal{L}_{\text{data}}$ and historical memory gradient $\nabla_\phi \mathcal{L}_{\text{mem}}$ align ($\theta < 90°$), accelerating convergence toward L_{min}. **(Right) Non-stationary environment:** Abrupt transitions cause $\nabla_\phi \mathcal{L}_{\text{data}}$ to pivot, forming an obtuse angle $\theta > 90°$ (between the yellow and blue lines) with the lagging $\nabla_\phi \mathcal{L}_{\text{mem}}$. A fixed τ yields negative gain (red arrow), whereas our dynamic τ mechanism truncates obsolete memory, realigning the update (green arrow) toward the new optimum L'_{min}.

Lemma 5.5 (Riemannian Inner Product Condition Based on Membrane Potential Memory)**.** *Under the premise that the temporal gradient correlation (Assumption 5.4) holds, we define the memory gradient of the SNN as the exponentially decayed accumulation of historical gradients, i.e.,* $\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)} = \sum_{k=1}^{t-1} \tau^k \nabla_\phi \mathcal{L}_{\text{data}}^{(t-k)}$*, where* $\tau \in (0, 1)$ *is the membrane potential retention constant of the neuron. Then, the expected Riemannian inner product between the current data gradient and the memory gradient is strictly greater than zero:*

$$\mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\right)\right] > 0. \tag{11}$$

Remark: Lemma 5.5 provides the necessary prerequisite for Theorem 5.6 to hold. Because the leak factor τ of the membrane potential naturally acts as a temporal smoothing variable, SNNs can implicitly capture the gradient correlation ρ within continuous trajectories. This induces an acceleration-like effect during parameter updates, ultimately allowing SNNs to achieve greater information gain than Artificial Neural Networks during the exploration process.

Subsequently, to quantify the value of this parameter update mechanism for policy exploration under Definition 5.3, we compare it with traditional ANNs that rely solely on $\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}$, leading to the following theorem:

Theorem 5.6 (Information Gain Advantage of Membrane Potential Accumulation)**.** *Let the information gain induced by the parameter update be measured by the KL divergence between the variational distributions, i.e.,* $I(\Delta\phi_t) = D_{\text{KL}}[q(\theta \mid \phi_t + \Delta\phi_t) \,\|\, q(\theta \mid \phi_t)]$*. Suppose the Fisher information matrix, which is the Hessian matrix of the KL divergence evaluated at* ϕ_t*, is denoted by* $H(\phi_t)$*. If the cosine similarity between the memory gradient and the data gradient on the Riemannian manifold defined by* H *is strictly positive:*

$$\cos \angle_H\left(\nabla \mathcal{L}_{\text{data}}^{(t)}, \nabla \mathcal{L}_{\text{mem}}^{(t)}\right) = \frac{\left(\nabla \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla \mathcal{L}_{\text{mem}}^{(t)}\right)}{\left\|\nabla \mathcal{L}_{\text{data}}^{(t)}\right\|_H \cdot \left\|\nabla \mathcal{L}_{\text{mem}}^{(t)}\right\|_H} > 0, \tag{12}$$

then the single-step information gain produced by the SNN is strictly greater than that of the ANN, i.e., $\Delta I_t = I_t^{\text{SNN}} - I_t^{\text{ANN}} > 0$.

Remark: In the temporal trajectories of reinforcement learning, policy gradients from adjacent time steps typically exhibit high temporal correlation. This implies that the current data gradient $\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}$ is highly likely to maintain directional consistency with the moving average of its historical gradients $\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}$, provided that Assumption 5.4 holds. In this scenario, the membrane potential accumulation mechanism of

Table 1: The differences of various indicators in different environments are shown. Training time is counted by log and energy estimation is realized based on appendix

Environment	Metrics	Method			
		DQN	DRQN	DSQN	OSFD-DQN
Pac-Man MediumClassic (r=1)	Standard Deviation ↓	**7.688**	9.068	10.303	10.615
	Mean Reward ↑	-123.383	-24.263	-141.269	**-12.644**
	Training Time (h) ↓	5.8	8.2	12.6	**3.7**
	Energy Consumption ↓ (μJ)	2.85	7.68	0.336	**0.084**
Mujoco Maze-S-shape (r=1)	Standard Deviation ↓	0.031	0.128	0.026	**0.003**
	Mean Reward ↑	0.945	0.650	0.927	**0.999**
	Training Time (h) ↓	3.1	5.9	9.7	**2.4**
	Energy Consumption (μJ) ↓	2.85	7.68	0.336	**0.084**

SNNs is mathematically equivalent to natural momentum in optimization algorithms. By preserving consistency with historical gradients, it significantly amplifies the network's exploration step size on the parameter manifold, thereby yielding an information gain that surpasses that of traditional ANNs.

However, the data gradient $\nabla_\phi \mathcal{L}^{(t)}_{\text{data}}$ may also become misaligned with the memory gradient $\nabla_\phi \mathcal{L}^{(t)}_{\text{mem}}$; when this occurs, it could diminish the overall gain or even result in negative gain. Because states in complex reinforcement learning environments often undergo abrupt transitions, the current data gradient $\nabla_\phi \mathcal{L}^{(t)}_{\text{data}}$ is highly susceptible to conflicting with the memory gradient $\nabla_\phi \mathcal{L}^{(t)}_{\text{mem}}$, which leads to information loss during the model's exploration process. Consequently, mitigating such gradient conflicts serves as a necessary prerequisite for Theorem 5.6 to hold. How to best ensure that the conditions of this theorem are satisfied will be further explored in Section 5.3.

In complex control tasks, the partial observability of the environment frequently induces state aliasing, rendering single-step observations insufficient for accurately predicting future dynamics. Drawing upon Information Bottleneck theory, we can qualitatively elucidate the theoretical advantage of SNNs over memoryless ANNs in state representation.

Proposition 5.7 (SNN Representation Advantage Based on Information Retention)**.** *Let $\psi_{\text{ANN}}(s_t)$ denote the memoryless representation of a current observation, and $\psi_{\text{SNN}}(\tau_t)$ denote the SNN representation integrating the historical trajectory $\tau_t = (s_1, a_1, \ldots, s_t)$. By encoding historical context via membrane dynamics, SNNs retain greater mutual information regarding the future state S_{t+1} than ANNs:*

$$I(\psi_{\text{SNN}}(\tau_t); S_{t+1} \mid a_t) \;\geq\; I(\psi_{\text{ANN}}(s_t); S_{t+1} \mid a_t) \tag{13}$$

Analysis: Memoryless ANN mappings, $\psi_{\text{ANN}}(s_t)$, inherently suffer from state aliasing, yielding high conditional uncertainty $H(S_{t+1} \mid \psi_{\text{ANN}})$ and irreducible Bellman residuals. Conversely, SNNs leverage membrane potentials to encode historical trajectories into a latent space. Following the Data Processing Inequality, this temporal integration strictly maintains or enhances mutual information regarding future states. By expanding the information capacity of the feature subspace, SNNs effectively reduce conditional entropy, fundamentally lowering the Bellman error bound and ensuring superior sample efficiency and stability in continuous control.

5.3 From Theoretical Information Gain to Lightweight Computational Proxy

Theorem 5.6 establishes that satisfying the Riemannian inner product condition $\big(\nabla_\phi \mathcal{L}^{(t)}_{\text{data}}\big)^\top H(\phi_t)\big(\nabla_\phi \mathcal{L}^{(t)}_{\text{mem}}\big) > 0$ maximizes the information gain. However, exactly computing the large-scale Hessian matrix $H(\phi_t)$ in SNNs incurs significant computational overhead, necessitating a lightweight proxy metric. As illustrated in Figure 3, in stationary environments, the data and memory gradients form an acute angle, where a larger time constant τ facilitates momentum accumulation to

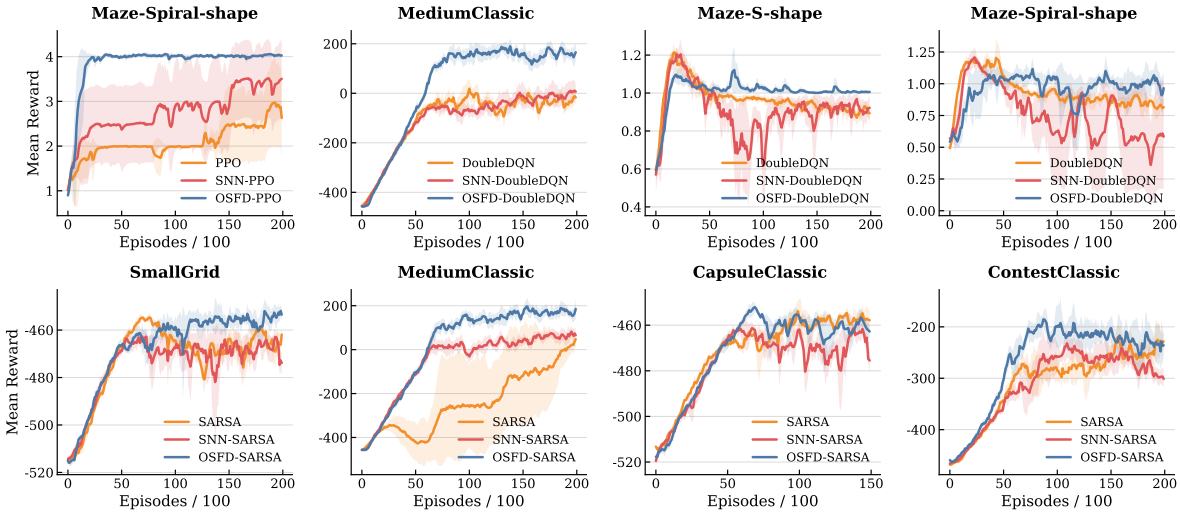

Figure 4: Performance comparison of various reinforcement learning algorithms across diverse environments. Solid lines represent the mean reward, while shaded regions denote the variance. Experimental results demonstrate that in the vast majority of tested environments, algorithms integrated with the OSFD framework exhibit higher final returns compared to the original baselines and their SNN counterparts.

accelerate convergence. Conversely, abrupt environmental transitions cause the data gradient to pivot swiftly while the memory gradient lags due to temporal inertia. This discrepancy creates an obtuse angle ($\theta > 90°$) between the two vectors. Under such non-stationary conditions, maintaining a fixed τ retains outdated gradient signals, yielding a negative gain. Thus, the network must rapidly truncate historical memory whenever $\theta > 90°$. Given that gradient conflicts induced by environmental shifts inherently manifest as deviations between adjacent latent state representations, we employ the feature cosine similarity as a computationally efficient proxy for the Riemannian gradient angle:

$$\cos \angle_H \left(\nabla \mathcal{L}_{\text{data}}^{(t)}, \nabla \mathcal{L}_{\text{mem}}^{(t)} \right) \propto \cos \left(\psi_{\text{SNN}}(s_t), \psi_{\text{SNN}}(s_{t-1}) \right) \tag{14}$$

Based on this proxy, we introduce an adaptive membrane potential leak mechanism to implicitly enforce the theoretical constraints without incurring additional forward computational overhead:

$$\tau_t = \tau_{\min} + (\tau_{\max} - \tau_{\min}) \cdot \sigma \Big(\beta \cdot \cos \left(\psi_{\text{SNN}}(s_t), \psi_{\text{SNN}}(s_{t-1}) \right) \Big) \tag{15}$$

where $\sigma(\cdot)$ is the Sigmoid function, β is the temperature coefficient, and $[\tau_{\min}, \tau_{\max}]$ delineates the physical stability boundaries. This mechanism dynamically reduces τ to discard obsolete memory during feature deviations, while maintaining a high τ otherwise, thereby ensuring sustained positive information gain.

6 Experiment

6.1 Enviroments And Experimental Settings

In this section, we evaluate our method on a set of benchmark environments, including Pacman Lipovetzky & Sardina (2018), MuJoCo Maze Xu et al. (2024) and DeepMind Lab Beattie et al. (2016). Our approach is implemented on top of several classical DRL algorithms to demonstrate the effectiveness of the underlying architecture and its compatibility with other reinforcement learning methods. We restrict the agent's observation area in both the Pacman and Maze tasks to simulate partially observable and dynamic environments. For instance, in Pacman, the agent must quickly avoid threats from ghosts while navigating obstacles and collecting food rewards—an example of real-time decision-making under survival pressure. In the Maze task, the agent is required to explore and locate sparse goal rewards, presenting an additional challenge of sparse

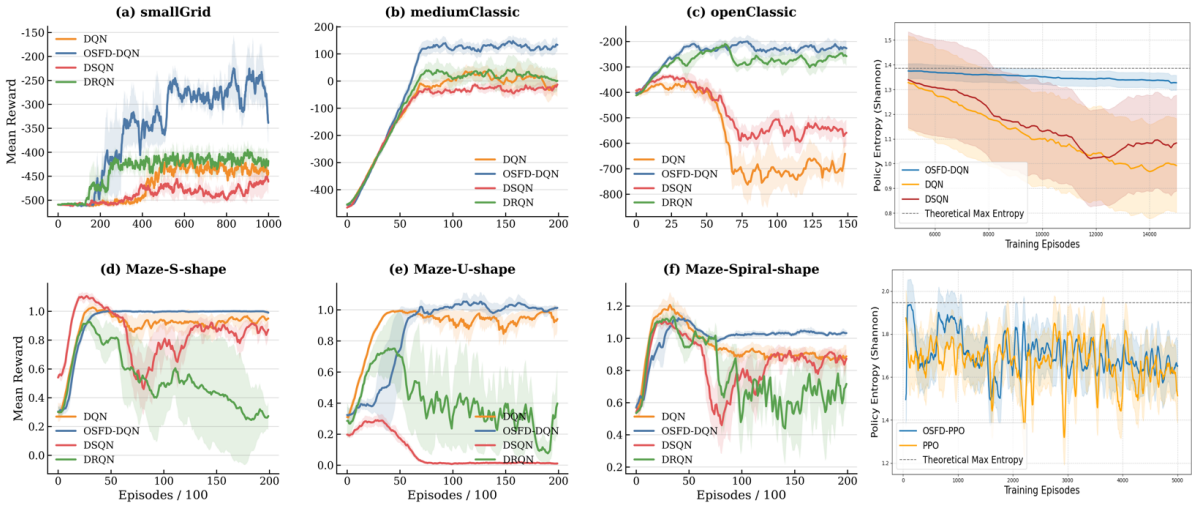

Figure 5: Performance and policy entropy dynamics of various algorithms across diverse environments. Subfigures (a)-(f) illustrate the mean reward learning curves of OSFD-DQN and the baseline algorithms. The two rightmost panels compare the evolution of policy entropy during training for the DQN variants (top right) and PPO variants (bottom right), respectively. Notably, in the Pacman environment, maintaining a relatively high level of policy entropy is typically required to effectively evade pursuit.

Table 2: Performance comparison of various PPO-based and DDQN-based methods across diverse environments, including four tasks from DeepMind Lab. Additionally, the table presents the impact of varying observation ranges on the final scores within the mediumClassic environment.

Domain	**Environment**	**PPO-based Methods**			**DDQN-based Methods**		
		PPO	**SNN-PPO**	**OSFD-PPO**	**DDQN**	**SNN-DDQN**	**OSFD-DDQN**
Mujoco-Maze	Maze-S-shape	2.374 ± 0.311	2.108 ± 0.470	**2.988 ± 0.012**	0.912 ± 0.009	0.944 ± 0.013	**1.010 ± 0.004**
	Maze-U-shape	1.640 ± 0.225	1.685 ± 0.129	**2.031** ± 0.078	0.951 ± 0.011	0.155 ± 0.042	**1.025 ± 0.008**
	Maze-spiral-shape	2.345 ± 0.545	3.137 ± 0.890	**4.013 ± 0.008**	0.887 ± 0.014	0.880 ± 0.027	**1.041 ± 0.003**
	Maze-square-random	0.672 ± 0.176	0.546 ± 0.118	**0.972 ± 0.008**	0.125 ± 0.009	0.112 ± **0.006**	**0.235** ± 0.008
	Maze-g1 (Success Rate %)	75.2 ± 12.8	68.5 ± 6.3	**94.5 ± 5.1**	48.5 ± 5.2	46.8 ± **4.1**	**74.5** ± 5.5
	Maze-g2 (Success Rate %)	62.6 ± 8.46	59.8 ± 10.7	**86.0** ± 7.2	39.2 ± 3.1	36.5 ± **2.0**	**61.2** ± 3.8
DeepMind Lab	Nav-maze-random-goal	23.42 ± 5.83	15.46 ± **1.05**	**25.77** ± 3.75	-	-	-
	Nav-maze-static	42.81 ± 18.51	37.73 ± 12.94	**51.20 ± 5.91**	-	-	-
	Seekavoid-arena	**65.65** ± 7.58	58.45 ± **2.06**	64.88 ± 2.21	-	-	-
	Stairway-to-melon	128.54 ± 21.92	115.69 ± 26.44	**142.24 ± 18.70**	-	-	-
Pac-Man	MediumClassic ($r = 1$)	643.75 ± 37.12	620.05 ± 22.89	**695.81** ± 35.06	-	-	-
	MediumClassic ($r = 2$)	745.06 ± 32.76	715.16 ± 25.53	**790.87** ± 5.80	-32.12 ± **4.55**	-13.13 ± 39.95	**153.74** ± 15.29
	MediumClassic ($r = 3$)	771.87 ± 18.88	753.89 ± 31.02	**778.94 ± 0.37**	-	-	-
	Contest	576.88 ± 13.06	558.26 ± **10.08**	**624.75** ± 15.46	-210.85 ± **1.20**	-250.60 ± 4.50	**-198.20** ± 6.50
	Original	825.41 ± 48.77	796.45 ± 35.99	**934.62** ± 28.65	-25.60 ± **2.80**	-45.20 ± 7.50	**102.30** ± 8.90
	Tricky	513.54 ± 21.51	483.65 ± 15.76	**615.05** ± 7.27	-135.20 ± **0.02**	-160.40 ± 8.50	**-95.40** ± 5.50

reward structure. Finally, DeepMind Lab introduces greater complexity by combining sparse rewards with continuously evolving environments, posing significant challenges for long-term planning and adaptation.

We apply the OSFD framework to various baseline algorithms and compare it against standard benchmark models. These include ANN-based baselines such as DQN, Double DQN, and PPO, as well as vanilla SNN-based methods such as DSQN Liu et al. (2022) and SNN-PPO. In environments with restricted observations, we further incorporate baselines featuring recurrent structures, such as DRQN Zeng et al. (2018), to comprehensively evaluate the advantages of our proposed method in terms of inference speed, energy consumption, and final performance. To ensure the robustness of our empirical results, we independently train each method in every environment using 10 distinct random seeds. Upon the completion of training, we evaluate the final

Table 3: Comprehensive evaluation of the proposed methods. **(a)** Performance comparison of methods in various environments and tasks. **(b)** In the ablation study of different parameter configurations, we observe that vanilla SNNs generally perform better with larger simulation time steps T, although their performance remains bounded by that of ANNs. Moreover, models equipped with residual potential accumulation consistently outperform those without it.

(a) Performance comparison in various environments

Domain	Environment	DQN	DRQN	DSQN	OSFD-DQN
Mujoco-Maze	Maze-S-shape	0.945 ± 0.031	0.650 ± 0.128	0.927 ± 0.026	**0.999 ± 0.003**
	Maze-U-shape	0.936 ± 0.015	0.309 ± 0.260	0.014 ± **0.001**	**1.021** ± 0.015
	Maze-spiral-shape	0.890 ± 0.017	0.668 ± 0.208	0.864 ± 0.016	**1.027 ± 0.004**
	Maze-square-random	0.102 ± 0.013	0.122 ± 0.024	0.088 ± **0.009**	**0.218** ± 0.012
	Maze-room	0.653 ± 0.175	0.732 ± 0.251	0.682 ± 0.124	**0.781 ± 0.081**
	Maze-g1 (Success Rate %)	41.3 ± 7.4	55.7 ± 12.5	38.6 ± **5.3**	**70.9** ± 7.8
	Maze-g2 (Success Rate %)	33.6 ± 4.0	42.1 ± 9.2	27.8 ± **2.6**	**57.4** ± 4.4
Pac-Man	SmallGrid	-436.44 ± **14.56**	-420.33 ± 27.93	-459.91 ± 33.74	**-272.91** ± 35.34
	MediumClassic	-11.63 ± 16.73	13.22 ± 58.29	-29.75 ± **4.63**	**125.97** ± 8.30
	Minimax	-492.82 ± **1.17**	-444.32 ± 55.49	-429.89 ± 3.94	**-406.38** ± 7.64
	OpenClassic	-678.94 ± 32.74	-295.74 ± 10.99	-536.77 ± **1.54**	**-267.03** ± 11.21
	Original	-42.05 ± **3.46**	**92.28** ± 16.59	-68.17 ± 9.49	88.78 ± 11.21
	Trapped	**-413.20** ± 126.93	-502.99 ± **0.022**	-485.41 ± 16.36	-446.57 ± 8.14
	Tricky	-151.06 ± **0.03**	-126.03 ± 23.54	-178.18 ± 11.03	**-109.05** ± 7.27
	Capsule (Base)	-470.64 ± **0.19**	-461.55 ± 35.66	-479.82 ± 2.79	**-458.41** ± 5.47
	+ Noise 0.1	-468.83 ± **1.78**	-461.63 ± 18.44	-469.05 ± 5.72	**-458.26** ±1.79
	+ Noise 0.3	-461.55 ± 0.53	**-442.72** ± 8.06	-455.99 ± **0.23**	-459.23 ± 0.62
	+ Noise 0.5	-492.57 ± 0.72	-522.10 ± 15.89	**-471.49**± **0.03**	-474.12 ± 1.16
	Contest (Base)	-225.40 ± **1.79**	**-193.22** ± 18.05	-271.35 ± 6.18	-215.57 ± 8.13
	+ Noise 0.1	-311.68 ± 8.36	**-276.00** ± 12.79	-355.32 ± 3.23	-333.67 ± **1.75**
	+ Noise 0.3	-417.87 ± 2.13	-432.11 ± 0.88	-450.30 ± 12.93	**-413.02 ± 0.56**
	+ Noise 0.5	-453.58 ± **0.40**	-513.79 ± 10.46	-435.2 ± 7.92	**-432.01** ± 9.25

(b) Ablation study of different parameter configurations

Environment	DSQN		OSFD-DQN	OSFD-DQN (w/o REPC)			OSFD-SARSA	
	$T = 2$	$T = 4$	w/REPC	$\tau = 0.1$	$\tau = 0.5$	$\tau = 0.9$	$\tau = 0.5$	w/REPC
Maze-S-shape	0.902	0.927	**0.999**	0.913	0.961	0.935	0.921	0.955
Maze-U-shape	0.013	0.014	**1.021**	0.841	0.952	0.901	0.890	0.931
Maze-spiral-shape	0.860	0.864	**1.027**	0.878	0.967	0.894	0.885	0.907
MediumClassic	-58.33 ± 8.09	-29.75 ± 4.63	125.97 ± 8.30	87.29 ± 4.88	108.41 ± 5.20	95.98 ± 13.45	155.46 ± 7.64	**173.07 ± 1.31**
Capsule	-502.15 ± 22.80	-479.82 ± **2.79**	**-458.41** ± 5.47	-512.87 ± 28.35	-472.04 ± 8.70	-496.00 ± 18.85	-491.51 ± 12.57	-478.51 ± 15.00
Original	-104.74 ± 11.86	-68.17 ± 9.49	**88.78** ± 11.21	72.11 ± 12.82	83.65 ± **4.06**	80.99 ± 4.50	46.06 ± 6.82	55.16 ± 15.05

learned policy by executing 20 independent test episodes. We report the mean and standard deviation of the achieved rewards and plot the complete learning curves derived from the training logs. Detailed experimental setups and hyperparameter configurations are provided in the Appendix.

6.2 Calculation Standard for Spiking Energy Consumption

In this section, by referring to the theoretical calculation methods for model energy consumption in spiking neural networks Qin et al. (2022), we quantify the computational overhead of different models to demonstrate the energy efficiency advantage of the proposed method in our experiments. In SNNs, the total number

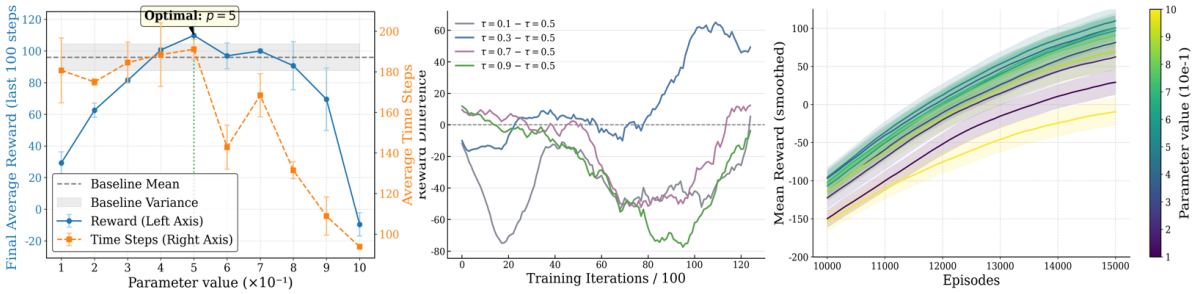

Figure 6: Dynamics of reward differences across training iterations under varying τ values. This illustrates the impact of different τ settings on the decision-making performance of the OSFD framework.

of Synaptic Operations (SOP) is commonly used as a metric, which measures the number of spike-based Accumulate (AC) operations. AC operations incur specific energy costs, the magnitude of which depends on the sparsity of spike firing. In contrast, ANNs use FLOPs(l) to represent the computational load of the l-th layer, corresponding to the number of floating-point Multiply-Accumulate (MAC) operations, which generally entail a significantly higher computational overhead.

The energy consumed per AC operation is $E_{AC} = 0.9$ pJ, whereas the energy consumed per MAC operation is $E_{MAC} = 4.6$ pJ. In SNNs, the total number of synaptic operations is typically defined as follows: $N_{AC} = \sum_{t=1}^{T} \sum_{l=1}^{L-1} \sum_{i=1}^{N^l} f_i^l \, s_i^l[t]$. For ANNs, the number of MAC operations is defined as: $N_{MAC} = \sum_{l=1}^{L-1} \sum_{i=1}^{N^l} f_i^l$. where L denotes the total number of layers in the network, N^l denotes the number of neurons in the l-th layer, and f_i represents the number of output connections for each neuron. $s_i^l[t]$ indicates the spike event of the i-th neuron in the l-th layer at time step t. Because neuron firing in spiking neural networks is generally sparse, we uniformly apply a spike firing rate of 15% in our calculations. Table 4 presents the statistics for the number of neuron connections in the linear and convolutional layers:

6.3 Performance and Results

We conducted experiments across multiple environments of varying difficulties in Pacman and MuJoCo Maze to evaluate the performance differences between various baseline algorithms adapted with the OSFD framework and their original counterparts. As illustrated in Figures 4 and 5, we compared algorithms implemented under both the vanilla SNN and vanilla ANN frameworks. Furthermore, we incorporated network architectures with recurrent structures as baselines to demonstrate the lightweight nature and performance advantages of our proposed method, which obviates the need for explicit recurrent structures.

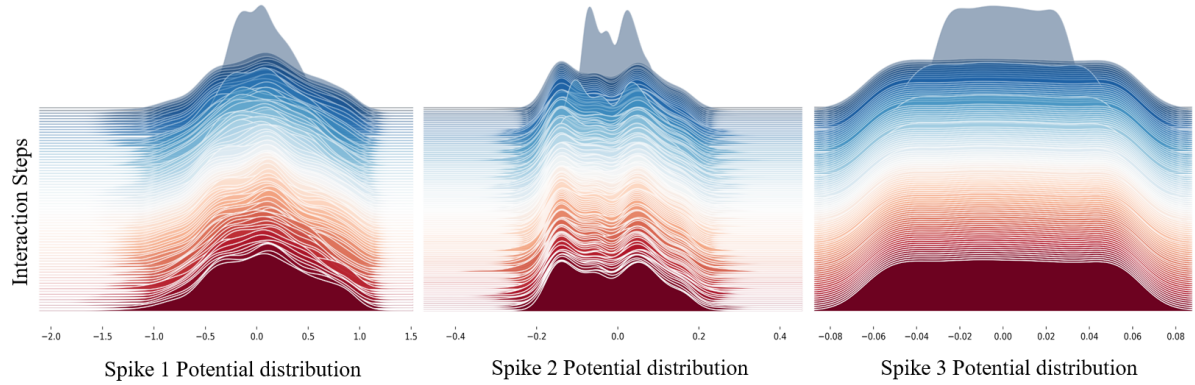

Figure 7: Dynamic evolution of spike potential distributions across varying interaction steps. Utilizing a ridge plot, the panels from left to right depict the progression of the potential distributions for Spike 1, Spike 2, and Spike 3, respectively, as the number of interaction steps increases.

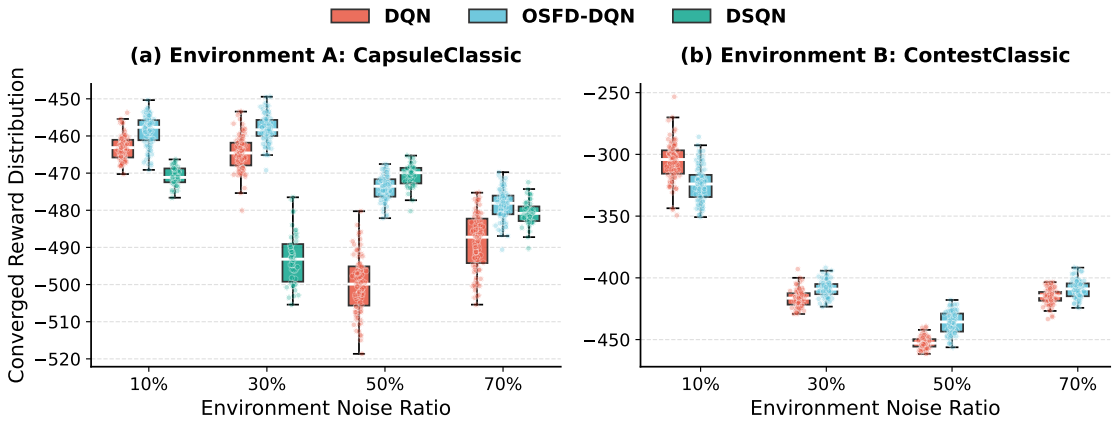

Figure 8: Analysis of algorithmic robustness across varying environmental noise ratios. The box plots illustrate the distribution of converged rewards for different algorithms in the (a) CapsuleClassic and (b) ContestClassic environments, evaluated under noise ratios ranging from 10% to 70%. Scatter points represent individual observations from independent trials.

Notably, the vanilla SNN-based methods utilize a simulation time step of $T = 4$, which empirically achieves near-optimal performance. As depicted in Figure 5, DRQN exhibits severe instability when deployed in sparse maze environments. Finally, Table 1 highlights the lightweight characteristics and computational agility of our approach.

We evaluated our method across several environments within DeepMind Lab. Table 2 demonstrates the empirical advantages of the PPO algorithm variants integrated with the OSFD architecture in these settings. Specifically, the `nav_maze_static` and `stairway_to_melon` environments are widely adopted to assess an agent's capacities for visual navigation, memory retention, and long-horizon planning. The `nav_maze_random_goal` environment is a sparse-reward maze navigation task with randomized goals, which imposes stringent demands on the exploration strategy. Furthermore, the `seekavoid_arena_01` task requires the agent to collect apples while actively avoiding lemons. Overall, our experimental results indicate that OSFD excels particularly in tasks necessitating long-horizon planning and short-term memory, all while demanding remarkably low computational energy consumption.

Table 2 presents an experimental analysis utilizing Double DQN as the baseline. Notably, OSFD-DDQN achieves the highest mean return compared to other Double Q-learning baselines. Furthermore, the policy entropy analysis in Figure 5 demonstrates that for agents in the Pacman environment, the OSFD architecture effectively prevents premature policy convergence to a narrow distribution, thereby enhancing the agent's ability to evade ghosts. Similarly, within DeepMind Lab, the OSFD architecture enables the agent to sustain a more robust exploratory capability across complex tasks.

Finally, we evaluate algorithmic robustness by introducing environmental noise ranging from 10% to 70% in Figure 8. In the `CapsuleClassic` and `ContestClassic` environments, as the noise ratio escalates, the performance of traditional baselines such as DQN degrades significantly, accompanied by a stark increase in variance. In contrast, OSFD-DQN exhibits exceptional noise tolerance across all noise levels; even under an extreme noise condition of 70%, it sustains a relatively high converged reward with minimal performance fluctuation. This provides compelling evidence that the OSFD architecture not only elevates foundational performance but also substantially bolsters the agent's robustness in uncertain environments.

6.4 Ablation Experiment

Effects of Information Gain-Based Residual Potential Accumulation Constraint (RPEC) In Table 3, we compare the performance differences across environments with and without RPEC, and also

report results under varying leakage factors. Clearly, RPEC mitigates decision accuracy loss, improves overall performance, and yields lower average standard deviation across 10 independent runs.

Different τ and Distribution of Potential As shown in Figure 7, the membrane potential exhibits layer-wise distributional differences: the deeper the layer, the more concentrated the distribution becomes, indicating that deeper layers encode increasingly salient features. In addition, in LIF neurons, both excessively large and overly small residual potential decay factors lead to noticeable performance degradation. As illustrated in Figure 6, when the curve with $\tau = 0.5$ is used as a reference, other decay values result in varying performance gaps across different iterations. The observed differences across steps highlight the necessity of dynamically tuning τ.

7 Conclusion and Discussion

In this work, we introduced a reinforcement learning framework based on one-spike decision (OSFD), aiming to provide a brain-inspired solution for large-scale deployment of agents in complex, dynamic real-world environments. Experimental results demonstrate that, compared to ANNs, the proposed OSFD framework based on spiking neural networks can accomplish tasks more efficiently with significantly lower computational cost. This advantage stems from the inherent temporal structure and biologically inspired membrane potential dynamics of SNNs, forming a unique computational paradigm distinct from conventional neural models. We conclude that SNNs may be intrinsically more suitable for reinforcement learning, and we believe this perspective offers new directions for future research. Despite its theoretical and empirical advantages, OSFD is currently an exploratory framework, it has the following limitations.

Limitations: During the sampling process in DRL, we need to store the residual membrane potential at each decision step. This increases the storage requirements of the replay buffer, leading to additional memory overhead. And the current strategy for residual potential accumulation is still relatively primitive. It relies on a simple additive accumulation modulated by a leakage factor τ. While this approach has proven to be empirically effective, it may not fully exploit the biological plausibility and representational power of spiking neural dynamics. We are actively exploring improved accumulation mechanisms to enhance the structural expressiveness and efficiency of residual potentials in ongoing work.

References

Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, et al. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

Jiahang Cao, Mingyuan Sun, Ziqing Wang, Hao Cheng, Qiang Zhang, Renjing Xu, et al. Spiking neural network as adaptive event stream slicer. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

Ding Chen, Peixi Peng, Tiejun Huang, and Yonghong Tian. Deep reinforcement learning with spiking q-learning. *arXiv preprint arXiv:2201.09754*, 2022.

Ding Chen, Peixi Peng, Tiejun Huang, and Yonghong Tian. Fully spiking actor network with intralayer connections for reinforcement learning. *IEEE Transactions on Neural Networks and Learning Systems*, 36 (2):2881–2893, 2024.

Jayabrata Chowdhury, Venkataramanan Shivaraman, Suresh Sundaram, and PB Sujit. Graph-based prediction and planning policy network (gp3net) for scalable self-driving in dynamic environments using deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11606–11614, 2024.

Sayeed Shafayet Chowdhury, Nitin Rathi, and Kaushik Roy. Towards ultra low latency spiking neural networks for vision and sequential tasks using temporal pruning. In *European Conference on Computer Vision*, pp. 709–726. Springer, 2022.

Gourav Datta, Zeyu Liu, Anni Li, and Peter A Beerel. Spiking neural networks with dynamic time steps for vision transformers. *arXiv preprint arXiv:2311.16456*, 2023.

Shikuang Deng, Yuhang Wu, Kangrui Du, and Shi Gu. Spiking token mixer: An event-driven friendly former structure for spiking neural networks. *Advances in Neural Information Processing Systems*, 37: 128825–128846, 2024.

Jianhao Ding, Zhaofei Yu, Yonghong Tian, and Tiejun Huang. Optimal ann-snn conversion for fast and accurate inference in deep spiking neural networks. *arXiv preprint arXiv:2105.11654*, 2021.

Yongqi Ding, Lin Zuo, Mengmeng Jing, Pei He, and Yongjun Xiao. Shrinking your timestep: Towards low-latency neuromorphic object recognition with spiking neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11811–11819, 2024.

Xuanyi Dong, Junshi Huang, Yi Yang, and Shuicheng Yan. More is less: A more complicated network with less inference complexity. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 5840–5848, 2017.

Xinyang Gu, Yen-Jen Wang, and Jianyu Chen. Humanoid-gym: Reinforcement learning for humanoid robot with zero-shot sim2real transfer. *arXiv preprint arXiv:2404.05695*, 2024.

Bing Han, Gopalakrishnan Srinivasan, and Kaushik Roy. Rmp-snn: Residual membrane potential neuron for enabling deeper high-accuracy and low-latency spiking neural network. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 13558–13567, 2020.

Jianye Hao, Tianpei Yang, Hongyao Tang, Chenjia Bai, Jinyi Liu, Zhaopeng Meng, Peng Liu, and Zhen Wang. Exploration in deep reinforcement learning: From single-agent to multiagent domain. *IEEE Transactions on Neural Networks and Learning Systems*, 2023a.

Zecheng Hao, Tong Bu, Jianhao Ding, Tiejun Huang, and Zhaofei Yu. Reducing ann-snn conversion error through residual membrane potential. In *Proceedings of the AAAI conference on artificial intelligence*, volume 37, pp. 11–21, 2023b.

Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Vime: Variational information maximizing exploration. *Advances in neural information processing systems*, 29, 2016.

Yangfan Hu, Qian Zheng, Xudong Jiang, and Gang Pan. Fast-snn: Fast spiking neural network by converting quantized ann. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(12):14546–14562, 2023.

Yulong Huang, Xiaopeng Lin, Hongwei Ren, Haotian Fu, Yue Zhou, Zunchang Liu, Biao Pan, and Bojun Cheng. Clif: Complementary leaky integrate-and-fire neuron for spiking neural networks. *arXiv preprint arXiv:2402.04663*, 2024.

Laurent Itti and Pierre Baldi. Bayesian surprise attracts human attention. *Advances in neural information processing systems*, 18, 2005.

Yizhou Jiang, Kunlin Hu, Tianren Zhang, Haichuan Gao, Yuqian Liu, Ying Fang, and Feng Chen. Spatio-temporal approximation: A training-free snn conversion for transformers. In *The Twelfth International Conference on Learning Representations*, 2024.

Napat Karnchanachari, Dimitris Geromichalos, Kok Seang Tan, Nanxiang Li, Christopher Eriksen, Shakiba Yaghoubi, Noushin Mehdipour, Gianmarco Bernasconi, Whye Kit Fong, Yiluan Guo, et al. Towards learning-based planning: The nuplan benchmark for real-world autonomous driving. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 629–636. IEEE, 2024.

Andrew Li, Zizhao Chen, Toryn Klassen, Pashootan Vaezipoor, Rodrigo Toro Icarte, and Sheila McIlraith. Reward machines for deep rl in noisy and uncertain environments. *Advances in Neural Information Processing Systems*, 37:110341–110368, 2024a.

Keqin Li, Jiajing Chen, Dezhi Yu, Tao Dajun, Xinyu Qiu, Jieting Lian, Ryan Ji, Shengyuan Zhang, Zhenyu Wan, Baiwei Sun, et al. Deep reinforcement learning-based obstacle avoidance for robot movement in warehouse environments. In *2024 IEEE 6th International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, pp. 342–348. IEEE, 2024b.

Yang Li and Yi Zeng. Efficient and accurate conversion of spiking neural network with burst spikes. *arXiv preprint arXiv:2204.13271*, 2022.

Yuhang Li, Tamar Geller, Youngeun Kim, and Priyadarshini Panda. Seenn: towards temporal spiking early exit neural networks. *Advances in Neural Information Processing Systems*, 36:63327–63342, 2023a.

Yuhang Li, Abhishek Moitra, Tamar Geller, and Priyadarshini Panda. Input-aware dynamic timestep spiking neural networks for efficient in-memory computing. In *2023 60th ACM/IEEE Design Automation Conference (DAC)*, pp. 1–6. IEEE, 2023b.

Nir Lipovetzky and Sebastian Sardina. Pacman capture the flag in ai courses. *IEEE Transactions on Games*, 11(3):296–299, 2018.

Guisong Liu, Wenjie Deng, Xiurui Xie, Li Huang, and Huajin Tang. Human-level control through directly trained deep spiking q-networks. *IEEE transactions on cybernetics*, 53(11):7187–7198, 2022.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Matthias Ochs, Markus Dietl, and Ralf Brederlow. An analog and time-discrete neuron with charge-injection for use in ultra-low power spiking neural networks. In *2024 19th Conference on Ph. D Research in Microelectronics and Electronics (PRIME)*, pp. 1–4. IEEE, 2024.

Katerina Maria Oikonomou, Ioannis Kansizoglou, and Antonios Gasteratos. A hybrid spiking neural network reinforcement learning agent for energy-efficient object manipulation. *Machines*, 11(2):162, 2023.

Guanchao Qiao, Ning Ning, Yue Zuo, Pujun Zhou, Mingliang Sun, Shaogang Hu, Qi Yu, and Yang Liu. Spatio-temporal fusion spiking neural network for frame-based and event-based camera sensor fusion. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 2024.

Lang Qin, Rui Yan, and Huajin Tang. A low latency adaptive coding spiking framework for deep reinforcement learning. *arXiv preprint arXiv:2211.11760*, 2022.

Nitin Rathi and Kaushik Roy. Lite-snn: Leveraging inherent dynamics to train energy-efficient spiking neural networks for sequential learning. *IEEE Transactions on Cognitive and Developmental Systems*, 2024.

Hangchi Shen, Qian Zheng, Huamin Wang, and Gang Pan. Rethinking the membrane dynamics and optimization objectives of spiking neural networks. *Advances in Neural Information Processing Systems*, 37: 92697–92720, 2024a.

Sicheng Shen, Dongcheng Zhao, Guobin Shen, and Yi Zeng. Tim: an efficient temporal interaction module for spiking transformer. *arXiv preprint arXiv:2401.11687*, 2024b.

Xinyu Shi, Jianhao Ding, Zecheng Hao, and Zhaofei Yu. Towards energy efficient spiking neural networks: An unstructured pruning framework. In *The Twelfth International Conference on Learning Representations*, 2024.

Xiaotian Song, Andy Song, Rong Xiao, and Yanan Sun. One-step spiking transformer with a linear complexity. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 3142–3150, 2024.

Guangzhi Tang, Neelesh Kumar, Raymond Yoo, and Konstantinos Michmizos. Deep reinforcement learning with population-coded spiking neural network for continuous control. In *Conference on Robot Learning*, pp. 2016–2029. PMLR, 2021.

Erwin Walraven, Joris Sijs, and Gertjan J Burghouts. Information gathering in pomdps using active inference. *Autonomous Agents and Multi-Agent Systems*, 39(1):1–22, 2025.

Zichun Xu, Yuntao Li, Xiaohang Yang, Zhiyuan Zhao, Lei Zhuang, and Jingdong Zhao. Open-source reinforcement learning environments implemented in mujoco with franka manipulator. In *2024 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, pp. 709–714. IEEE, 2024.

Luca Zanatta, Francesco Barchi, Simone Manoni, Silvia Tolu, Andrea Bartolini, and Andrea Acquaviva. Exploring spiking neural networks for deep reinforcement learning in robotic tasks. *Scientific Reports*, 14 (1):30648, 2024.

Jinghong Zeng, Jianming Hu, and Yi Zhang. Adaptive traffic signal control with deep recurrent q-learning. In *2018 IEEE intelligent vehicles symposium (IV)*, pp. 1215–1220. IEEE, 2018.

A Proofs of Theoretical Results

A.1 Proof of Lemma 5.1

For ease of presentation, we first restate the theorem and then introduce its proof.

Lemma A.1 (Mutual Information-Driven Policy Incentive) In this setting, we maintain a model of the environmental dynamics $p(s_{t+1}|s_t, a_t, \theta)$ and treat the unknown model parameters as a random variable Θ, with its specific realization being $\theta \in \Theta$. Given the agent's history up to time step t, denoted as $\xi_t = \{s_1, a_1, \ldots, s_t\}$, and the current action a_t, we define the information gain for exploration as the conditional mutual information between the next state S_{t+1} and the model parameters Θ. According to information theory:

$$I(S_{t+1}; \Theta|\xi_t, a_t) = \mathbb{E}_{s_{t+1},\theta\sim p(s_{t+1},\theta|\xi_t,a_t)}\left[\log \frac{p(s_{t+1}, \theta|\xi_t, a_t)}{p(s_{t+1}|\xi_t, a_t)p(\theta|\xi_t, a_t)}\right] \tag{16}$$

This mutual information can be equivalently expanded as the difference in conditional entropy, or the expected KL divergence between the posterior and the prior:

$$\begin{aligned} I(S_{t+1}; \Theta|\xi_t, a_t) &= H(\Theta|\xi_t, a_t) - H(\Theta|S_{t+1}, \xi_t, a_t) \\ &= \mathbb{E}_{s_{t+1}\sim p(s_{t+1}|\xi_t,a_t)}\Big[D_{\mathrm{KL}}[p(\theta|\xi_t, a_t, s_{t+1}) \,\|\, p(\theta|\xi_t, a_t)]\Big] \end{aligned} \tag{17}$$

Proof A.1: This proof aims to elucidate why information gain can be equivalently reformulated as the expectation of Kullback-Leibler (KL) divergence, thereby providing a theoretical foundation for the derivation of subsequent methods and conclusions. First, consider the transition dynamics model of the environment $p(s_{t+1} \mid s_t, a_t, \theta)$, where the model parameters are treated as a random variable Θ.

From information theory, the conditional mutual information between the parameters Θ and the next state S_{t+1} can be defined as the difference between the prior entropy and the conditional entropy:

$$I(S_{t+1}; \Theta \mid \xi_t, a_t) = H(\Theta \mid \xi_t, a_t) - H(\Theta \mid \xi_t, a_t, S_{t+1}) \tag{18}$$

where $\xi_t = \{s_1, a_1, \ldots, s_t\}$ denotes the historical trajectory up to time step t.

By the definition of mutual information, this can be expanded into the expectation of the log-ratio between the joint distribution and the product of its marginal distributions:

$$I(S_{t+1}; \Theta \mid \xi_t, a_t) = \mathbb{E}_{(s_{t+1},\theta)\sim p(s_{t+1},\theta|\xi_t,a_t)}\left[\log \frac{p(s_{t+1}, \theta \mid \xi_t, a_t)}{p(s_{t+1} \mid \xi_t, a_t)\, p(\theta \mid \xi_t, a_t)}\right] \tag{19}$$

Applying Bayes' theorem, the joint probability can be factorized as $p(s_{t+1}, \theta \mid \xi_t, a_t) = p(\theta \mid \xi_t, a_t, s_{t+1})\, p(s_{t+1} \mid \xi_t, a_t)$. Substituting this into the above equation and canceling out $p(s_{t+1} \mid \xi_t, a_t)$ in the denominator, the expectation can be decomposed into nested expectations with respect to s_{t+1} and θ:

$$I(S_{t+1}; \Theta \mid \xi_t, a_t) = \mathbb{E}_{s_{t+1}\sim p(s_{t+1}|\xi_t,a_t)}\left[\mathbb{E}_{\theta\sim p(\theta|\xi_t,a_t,s_{t+1})}\left[\log \frac{p(\theta \mid \xi_t, a_t, s_{t+1})}{p(\theta \mid \xi_t, a_t)}\right]\right] \tag{20}$$

Noting that the inner expectation corresponds exactly to the definition of the Kullback-Leibler divergence between two probability distributions, this ultimately simplifies to:

$$I(S_{t+1}; \Theta \mid \xi_t, a_t) = \mathbb{E}_{s_{t+1}\sim p(s_{t+1}|\xi_t,a_t)}\Big[D_{\mathrm{KL}}\big[p(\Theta \mid \xi_t, a_t, s_{t+1}) \,\|\, p(\Theta \mid \xi_t, a_t)\big]\Big] \tag{21}$$

A.2 Proof of Lemma 5.2

For ease of presentation, we first restate the theorem and then introduce its proof.

Lemma A.2. (Variational Bayesian Posterior Approximation and Information Gain Metric) Let $q(\theta;\,\phi)$ be the variational distribution used to approximate the intractable posterior $p(\theta \mid \mathcal{D})$, and assume it is a fully factorized Gaussian distribution,$\mathcal{D} = \{\xi_t, a_t, s_{t+1}\}$ to represent the current observation:

$$q(\theta;\,\phi) = \prod_{i=1}^{|\theta|} \mathcal{N}(\theta_i \mid \mu_i, \sigma_i^2), \qquad \phi = \{\mu, \sigma\}. \tag{22}$$

Minimizing the KL divergence between $q(\theta;\,\phi)$ and the true posterior is equivalent to maximizing the Evidence Lower Bound (ELBO), whose objective function can be expressed as:

$$\mathcal{L}[q(\theta;\,\phi), \mathcal{D}] = \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D} \mid \theta)] - D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta)]. \tag{23}$$

During the policy update process, for any given parameter update step $\Delta\phi$, the KL divergence between the variational distributions before and after the update—which represents the induced intrinsic information gain—can be approximated via a second-order Taylor expansion using the Fisher information matrix:

$$D_{\mathrm{KL}}[q(\theta;\,\phi_t + \Delta\phi) \,\|\, q(\theta;\,\phi_t)] \approx \frac{1}{2}\Delta\phi^\top H(\phi_t)\Delta\phi, \tag{24}$$

where $H(\phi_t) = \nabla_\phi^2 D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, q(\theta;\phi_t)]\big|_{\phi=\phi_t}$ is the Hessian matrix under the current parameters.

Proof A.2: This proof provides the derivation for the Evidence Lower Bound (ELBO) objective as stated in the lemma. By measuring the divergence between the variational distribution $q(\theta;\phi)$ and the true posterior $p(\theta \mid \mathcal{D})$, we have:

$$D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta \mid \mathcal{D})] = \mathbb{E}_{q(\theta;\phi)}\left[\log \frac{q(\theta;\phi)}{p(\theta \mid \mathcal{D})}\right]. \tag{25}$$

Applying Bayes' theorem, $p(\theta \mid \mathcal{D}) = \frac{p(\mathcal{D}|\theta)p(\theta)}{p(\mathcal{D})}$, to the equation above and expanding the logarithmic term, we obtain:

$$\begin{aligned} D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta \mid \mathcal{D})] &= \mathbb{E}_{q(\theta;\phi)}\left[\log \frac{q(\theta;\phi)p(\mathcal{D})}{p(\mathcal{D} \mid \theta)p(\theta)}\right] \\ &= \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D})] - \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D} \mid \theta)] + \mathbb{E}_{q(\theta;\phi)}\left[\log \frac{q(\theta;\phi)}{p(\theta)}\right]. \end{aligned} \tag{26}$$

Since $\log p(\mathcal{D})$ is independent of θ, it can be factored out of the expectation. The final term corresponds exactly to the KL divergence between the variational distribution and the prior. Thus, this yields:

$$D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta \mid \mathcal{D})] = \log p(\mathcal{D}) - \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D} \mid \theta)] + D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta)]. \tag{27}$$

Rearranging the terms and leveraging the non-negativity of the KL divergence ($D_{\mathrm{KL}} \geq 0$), we obtain the lower bound for the log-evidence:

$$\log p(\mathcal{D}) \geq \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D} \mid \theta)] - D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta)]. \tag{28}$$

Consequently, to encourage $q(\theta;\phi)$ to closely approximate the true posterior, the variational lower bound $\mathcal{L}[q(\theta;\phi), \mathcal{D}]$ that must be maximized is given by:

$$\mathcal{L}[q(\theta;\phi), \mathcal{D}] = \mathbb{E}_{q(\theta;\phi)}[\log p(\mathcal{D} \mid \theta)] - D_{\mathrm{KL}}[q(\theta;\phi) \,\|\, p(\theta)]. \tag{29}$$

A.3 Proof of Lemma 5.5

For ease of presentation, we first restate the theorem and then introduce its proof.

Lemma A.3. (Riemannian Inner Product Condition Based on Membrane Potential Memory) Under the premise that the temporal gradient correlation (Assumption 5.4) holds, we define the memory gradient of the SNN as the exponentially decayed accumulation of historical gradients, i.e., $\nabla_\phi \mathcal{L}_{\mathrm{mem}}^{(t)} = \sum_{k=1}^{t-1} \tau^k \nabla_\phi \mathcal{L}_{\mathrm{data}}^{(t-k)}$,

where $\tau \in (0,1)$ is the membrane potential retention constant of the neuron. Then, the expected Riemannian inner product between the current data gradient and the memory gradient is strictly greater than zero:

$$\mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\right)\right] > 0. \tag{30}$$

Proof A.3: For ease of presentation, we first restate the theorem and then introduce its proof. Substituting the expansion of the SNN memory gradient into the expected Riemannian inner product and leveraging the linearity of expectation, we obtain:

$$\begin{aligned}\mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\right)\right] &= \mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\sum_{k=1}^{t-1}\tau^k \nabla_\phi \mathcal{L}_{\text{data}}^{(t-k)}\right)\right] \\ &= \sum_{k=1}^{t-1}\tau^k\,\mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t-k)}\right)\right].\end{aligned} \tag{31}$$

Applying the lower bound from Assumption 5.4 to the equation above yields:

$$\mathbb{E}\left[\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\right)\right] \geq \sum_{k=1}^{t-1}\tau^k(\rho^k C) = C\sum_{k=1}^{t-1}(\tau\rho)^k. \tag{32}$$

Given that $C > 0$, $\tau \in (0,1)$, and $\rho \in (0,1)$, it follows that their product $\tau\rho \in (0,1)$. Consequently, the sum of this geometric series is strictly positive. Therefore, the expected Riemannian inner product is strictly greater than 0, which concludes the proof.

A.4 Proof of Theorem 5.6

For ease of presentation, we first restate the theorem and then introduce its proof.

Theorem A.4. (Information Gain Advantage of Membrane Potential Accumulation) Let the information gain induced by the parameter update be measured by the KL divergence between the variational distributions, i.e., $I(\Delta\phi_t) = D_{\text{KL}}[q(\theta \mid \phi_t + \Delta\phi_t) \,\|\, q(\theta \mid \phi_t)]$. Suppose the Fisher information matrix, which is the Hessian matrix of the KL divergence evaluated at ϕ_t, is denoted by $H(\phi_t)$. If the cosine similarity between the memory gradient and the data gradient on the Riemannian manifold defined by H is strictly positive:

$$\cos\angle_H\left(\nabla\mathcal{L}_{\text{data}}^{(t)}, \nabla\mathcal{L}_{\text{mem}}^{(t)}\right) = \frac{\left(\nabla\mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla\mathcal{L}_{\text{mem}}^{(t)}\right)}{\left\|\nabla\mathcal{L}_{\text{data}}^{(t)}\right\|_H \cdot \left\|\nabla\mathcal{L}_{\text{mem}}^{(t)}\right\|_H} > 0, \tag{33}$$

then the single-step information gain produced by the SNN is strictly greater than that of the ANN, i.e., $\Delta I_t = I_t^{\text{SNN}} - I_t^{\text{ANN}} > 0$.

Proof A.4: Based on the fundamental theorem of information geometry, we perform a second-order Taylor expansion of the Kullback-Leibler (KL) divergence around the current parameter ϕ_t. Since the KL divergence achieves its minimum value of 0 when $\Delta\phi = \mathbf{0}$, the zeroth-order and first-order gradient terms vanish, yielding the following quadratic approximation:

$$I(\Delta\phi_t) = D_{\text{KL}}[q(\theta \mid \phi_t + \Delta\phi_t) \,\|\, q(\theta \mid \phi_t)] \approx \frac{1}{2}\Delta\phi_t^\top H(\phi_t)\Delta\phi_t. \tag{34}$$

For Artificial Neural Networks (ANNs), the parameter update relies exclusively on the current data, i.e., $\Delta\phi_t^{\text{ANN}} = \nabla_\phi \mathcal{L}_{\text{data}}^{(t)}$. The corresponding information gain can be approximated as:

$$I_t^{\text{ANN}} \approx \frac{1}{2}\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right)^\top H(\phi_t)\left(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\right). \tag{35}$$

For Spiking Neural Networks (SNNs), substituting the update rule $\Delta\phi_t^{\text{SNN}} = \nabla_\phi \mathcal{L}_{\text{data}}^{(t)} + \lambda \nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}$ into Equation 34 yields:

$$\begin{aligned} I_t^{\text{SNN}} &\approx \frac{1}{2}\big(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)} + \lambda \nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big)^\top H(\phi_t)\big(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)} + \lambda \nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big) \\ &= I_t^{\text{ANN}} + \lambda\big(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\big)^\top H(\phi_t)\big(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big) + \frac{\lambda^2}{2}\big(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big)^\top H(\phi_t)\big(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big). \end{aligned} \tag{36}$$

The difference in information gain between the two approaches is given by:

$$\Delta I_t = I_t^{\text{SNN}} - I_t^{\text{ANN}} = \lambda\big(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\big)^\top H(\phi_t)\big(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big) + \frac{\lambda^2}{2}\big\|\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big\|_H^2. \tag{37}$$

Since the Fisher information matrix $H(\phi_t)$ is positive semi-definite, the quadratic term is guaranteed to be non-negative, i.e., $\frac{\lambda^2}{2}\|\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\|_H^2 \geq 0$. Therefore, as long as the Riemannian inner product of the cross term is strictly positive, $\big(\nabla_\phi \mathcal{L}_{\text{data}}^{(t)}\big)^\top H(\phi_t)\big(\nabla_\phi \mathcal{L}_{\text{mem}}^{(t)}\big) > 0$ (which corresponds to Equation 33 holding true), it logically follows that $\Delta I_t > 0$. This completes the proof.

B Algorithm Details

Algorithm 1 OSFD Reinforcement Learning Framework (PPO Implementation)

Input: Discount factor γ; learning rates $\alpha_{\text{actor}}, \alpha_{\text{critic}}$; membrane potential decay range $\tau_{\min}, \tau_{\max}$; LIF parameters (m_n) $(n = 1, \ldots, N)$; replay buffer D; batch size B.

Output: Optimized policy π_ϕ, value function V_ψ

Initialize policy π_ϕ, value network V_ψ; Initialize membrane potentials $(m_n^1) = 0$, set $\tau \leftarrow 0.5$

repeat

{Interaction Phase}

for each time step t **do**

Input the observed state s_t into the network for single-timestep decision-making, and select action $a_t \sim \pi_\phi(a_t|s_t)$

Execute action a_t, and obtain reward r_t and next state s_{t+1}

Update the membrane potential m_t, continuously accumulating it across adjacent time steps

Compute the information gain $\cos\big(\psi_{\text{SNN}}(s_t), \psi_{\text{SNN}}(s_{t-1})\big)$

Update the membrane decay factor:

$$\tau_t = \tau_{\min} + (\tau_{\max} - \tau_{\min}) \cdot \sigma\Big(\beta \cdot \cos\big(\psi_{\text{SNN}}(s_t), \psi_{\text{SNN}}(s_{t-1})\big)\Big)$$

Store transition $\{s_t, a_t, r_t, s_{t+1}, (\text{mem})_t\}$ into buffer D

end for

Reset membrane potentials $(m_n)_t = 0$, and set $\tau \leftarrow 0.5$

{Training Phase}

Sample a mini-batch B from D and load the corresponding membrane potentials

for each update step **do**

Compute advantages $A(s_t, a_t)$ and target returns $\hat{V}(s_t)$

Policy loss:

$$L_\pi = -\mathbb{E}_B[\min(r_t A_t, \text{clip}(r_t) A_t)]$$

Value loss:

$$L_V = \mathbb{E}_B[(r_t + \gamma V(s') - V(s))^2]$$

Update policy: $\phi \leftarrow \phi - \alpha_{\text{actor}} \nabla_\phi L_\pi$

Update value: $\psi \leftarrow \psi - \alpha_{\text{critic}} \nabla_\psi L_V$

end for

until convergence

C Experimental Details and Additional Results

C.1 Environment Settings

In this section, we present the detailed settings of our experiments. Figures 9 respectively illustrate selected environments from Pacman, Mujoco-Maze and DeepMind Lab. We aim to demonstrate the effectiveness of our method across a wide range of dynamic and complex scenarios. These include partially observable settings, sparse reward structures, and continuous state spaces, covering diverse challenges faced in real-world decision-making tasks. In both the Pacman and Mujoco-Maze environments, we restrict the agent's

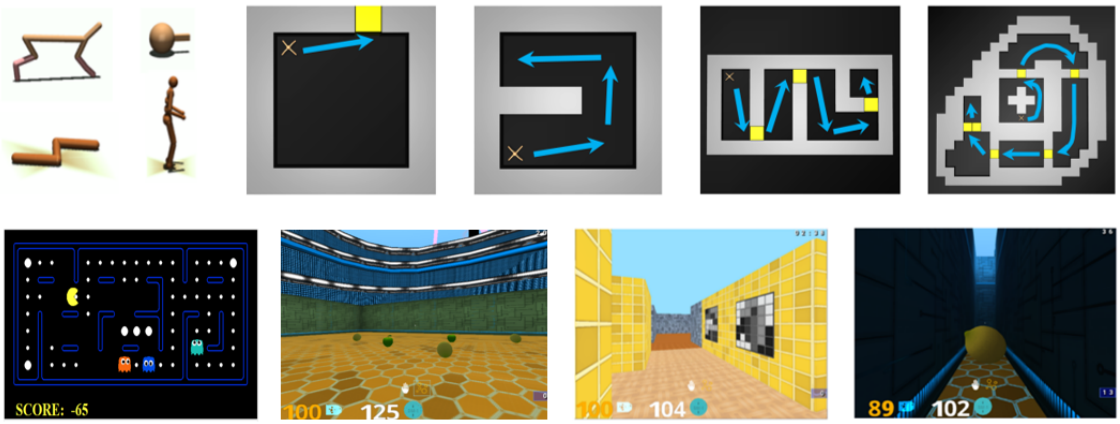

Figure 9: Our various experimental environments and details

observation range to better emulate natural intelligence systems. Specifically, a partial observation area is defined as a circular region of radius r, centered at the agent's current position. The observation field dynamically changes as the agent moves through the environment. In addition, we investigate how varying the observation radius r influences the performance of our method. In the Mujoco-Maze tasks, we design a series of maze-based environments in which the agent must navigate through doors to obtain rewards under partial observability. A local observation grid is constructed by discretizing the continuous Mujoco state space, and the observation is likewise centered around the agent (robot) to match the assumptions of our algorithm. To ensure robustness of the evaluation, we conduct experiments with 10 different random seeds for each method in every environment. Furthermore, we evaluate the final trained models using 20 independent test runs. We report the mean and standard deviation of the obtained scores or rewards, and we extract complete learning curves from the training logs.

C.2 Energy consumption calculation

In this section, we refer to the theoretical calculation method of model fiber in spiking neural networks Qin et al. (2022) to count the theoretical fiber of different models to demonstrate the fiber advantage of our method in experiments. In SNN, total synaptic operations (SOP) are usually used to describe fiber, which is the number of spike-based accumulation (AC) operations. AC has a large amount of power consumption and varies with the sparsity of spikes. The calculation in ANN is FLOPs(l), which refers to l floating-point operations, that is, the number of multiplication-accumulation (MAC) operations, which has more expensive calculations. The fiber of each AC calculation is $E_{AC} = 0.9pJ$ and the fiber of MAC is $E_{MAC} = 4.6pJ$. In SNN, we define the total number of synaptic operations as:

$$N_{AC} = \sum_{t=1}^{T} \sum_{l=1}^{L-1} \sum_{i=1}^{N^l} f_i^l s_i^l[t] \tag{38}$$

For ANN, it is defined as:

$$N_{MAC} = \sum_{l=1}^{L-1} \sum_{i=1}^{N^l} f_i^l \tag{39}$$

L represents the total number of layers, N^l denotes the number of neurons in layer l, f_{out} represents the number of outgoing connections for each neuron, $s_t^l[j]$ denotes the output spike of the j-th neuron in layer l at time step t.

The statistics of the number of neuron connections in the linear layer and convolutional layer are shown in the following table:

Table 4: FLOPs of convolutional layer l in ANN and SNN models

Model	**FLOPs of Convolutional Layer** l	
	Variable	**Value**
ANN	FL^l_{ANN}	$(k^l)^2 \times H_o^l \times W_o^l \times C_o^l \times C_i^l$
SNN	FL^l_{SNN}	$(k^l)^2 \times H_o^l \times W_o^l \times C_o^l \times C_i^l \times \zeta^l$

C.3 Hyperparameters Details

Table 6 illustrates the specific formulations of the OSFD framework when applied to the representative off-policy algorithm, DQN, and the on-policy architecture, PPO. It also details the hyperparameter configurations for both the proposed experimental methods and the baseline algorithms evaluated in this section. Specifically, the fundamental configurations of our methods remain consistent with the ANN baselines, with the primary distinction being the substitution of ReLU activation functions with LIF neurons. The experimental networks consist of multiple convolutional and linear layers, establishing a standard deep neural network architecture. Furthermore, this work introduces an observation radius parameter, r, in the Pacman and MuJoCo Maze environments. By adjusting the magnitude of r to restrict the agent's observation range, we investigate the effectiveness of the proposed method under Partially Observable Markov Decision Process (POMDP) settings.

Table 5: Membrane State (`mem`) Loading Strategy in OSFD-DQN

Stage	**Network**	**Membrane Handling Strategy**
Training	Policy Net	Load `mem` from current state before each update
	Target Net	Load corresponding `mem` from previous step
Inference (per step)	Policy Net	Load `mem` from previous timestep: $\texttt{mem}_{t-1} \rightarrow \texttt{mem}_t$
	Target Net	Not used during inference
Episode End		Reset all `mem`: $\texttt{mem} \leftarrow 0$

Table **??** summarizes the control logic for membrane potential states within the OSFD framework. During the training phase, both the policy network and the target network load the corresponding historical membrane potential states to ensure temporal consistency. During the inference phase, only the policy network propagates membrane potential states across time steps. At the conclusion of each episode, the membrane potentials are reset to zero. The advantage of this mechanism lies in maintaining the consistency of membrane potentials during both the sampling and training processes. This not only facilitates better convergence for the reinforcement learning algorithms but also maximizes the performance benefits derived from residual potentials.

C.4 Additional Results

This figure 10 illustrates the kernel density estimation (KDE) distributions of neuronal membrane potentials across three distinct network configurations or temporal states (‘mem1‘, ‘mem2‘, and ‘mem3‘) within a

Table 6: Hyperparameter and network architecture comparisons between ANN and OSFD under DQN (grid input) and PPO (image input) algorithms.

Parameter	ANN-DQN	OSFD-DQN	ANN-PPO	OSFD-PPO
Hyperparameters				
Discount factor γ	0.95	0.95	0.98	0.98
Learning rate	0.0002	0.0002	Actor: 1×10^{-4} Critic: 1×10^{-3}	Actor: 1×10^{-4} Critic: 1×10^{-3}
Batch size	32	32	32	32
Replay buffer capacity	50000	50000	–	–
Target network update freq.	100 steps	100 steps	–	–
Epsilon (final / decay steps)	0.1 / 7500 steps	0.1 / 7500 steps	–	–
Initial training steps	300	300	–	–
Evaluation episodes per test	5	5	–	–
Observation radius r	1–4	1–4	–	–
GAE coefficient λ	–	–	0.95	0.95
PPO Epochs	–	–	10	10
Clip parameter ϵ	–	–	0.2	0.2
Total episodes	30000	30000	30000	30000
Hidden layer dimension	512	512	512	512
Episode length	–	–	1000 steps	1000 steps
Network Architecture				
Architecture type	2 Conv + 2 FC	2 Conv + 2 FC	3 Conv + 2 FC	3 Conv + 2 FC
Convolutional layers	Conv1: 3×3, 32, stride 1 Conv2: 2×2, 64, stride 1	Conv1: 3×3, 32, stride 1 Conv2: 2×2, 64, stride 1	Conv1: 8×8, 32, stride 4 Conv2: 4×4, 64, stride 2 Conv3: 3×3, 64, stride 1	Conv1: 8×8, 32, stride 4 Conv2: 4×4, 64, stride 2 Conv3: 3×3, 64, stride 1
Fully connected layers	FC1: 256 → 512 FC2: 512 → Action dim	FC1: 256 → 512 FC2: 512 → Action dim	FC1: 3136 → 512 FC2: 512 → Action dim	FC1: 3136 → 512 FC2: 512 → Action dim
Activation function	ReLU	LIF	ReLU	LIF
LIF parameters	–	Threshold: 1.0 τ: Dynamically computed γ: 1.0	–	Threshold: 1.0 τ: Dynamically computed γ: 1.0

spiking neural network (SNN). The left panel depicts a highly active state characterized by a broad potential distribution that progressively shifts toward depolarization; notably, ‘mem3‘ forms a high-density peak within the 0 to 1 interval, indicating robust forward spike firing as numerous neurons approach the activation threshold. Conversely, the middle panel demonstrates a restricted dynamic regime where the membrane potential range is significantly compressed. In this scenario, the ‘mem3‘ distribution becomes sharply con-

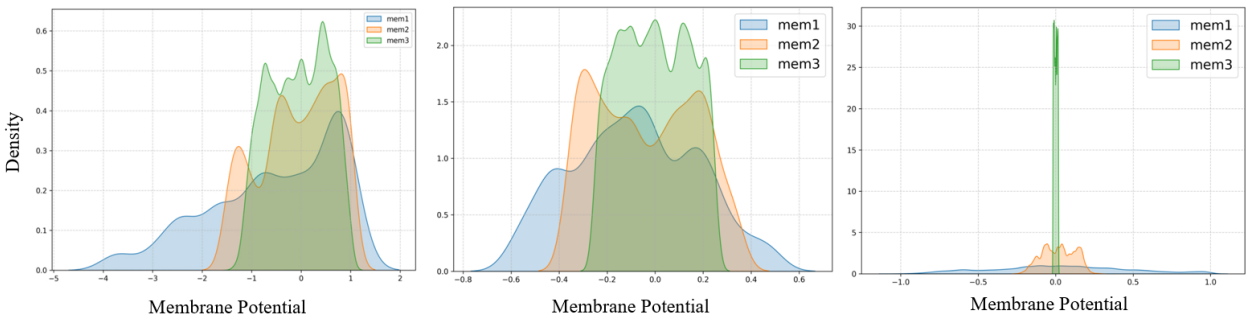

Figure 10: This figure illustrates the kernel density estimation (KDE) distributions of neuronal membrane potentials across three distinct network configurations or temporal states (‘mem1‘, ‘mem2‘, and ‘mem3‘) within a spiking neural network (SNN).

Table 7: Performance comparison of Double Q-learning based methods in various environments and tasks

Domain	Environment	DDQN	DDRQN	SNN-DDQN	OSFD-DDQN
Mujoco-Maze	Maze-S-shape	0.912 ± 0.009	0.725 ± 0.085	0.944 ± 0.013	**1.010 ± 0.004**
	Maze-U-shape	0.951 ± 0.011	0.412 ± 0.180	0.155 ± 0.042	**1.025 ± 0.008**
	Maze-spiral-shape	0.887 ± 0.014	0.740 ± 0.152	0.880 ± 0.027	**1.041 ± 0.003**
	Maze-square-random	0.125 ± 0.009	0.148 ± 0.015	0.112 ± **0.006**	**0.235** ± 0.008
	Maze-room	0.695 ± 0.120	0.768 ± 0.185	0.725 ± 0.095	**0.812 ± 0.065**
	Maze-g1 (Success Rate %)	48.5 ± 5.2	61.2 ± 8.5	46.8 ± **4.1**	**74.5** ± 5.5
	Maze-g2 (Success Rate %)	39.2 ± 3.1	48.6 ± 6.8	36.5 ± **2.0**	**61.2** ± 3.8
Pac-Man	SmallGrid	-410.25 ± **10.15**	-395.12 ± 18.50	-425.60 ± 22.14	**-255.40** ± 28.50
	MediumClassic	-32.12 ± **4.55**	28.55 ± 40.20	-13.13 ± 39.95	**153.74** ± 15.29
	Capsule	-465.10 ± **0.15**	-452.30 ± 25.40	-472.15 ± 1.95	**-450.25** ± 4.20
	Contest	-210.85 ± **1.20**	**-180.45** ± 12.30	-250.60 ± 4.50	-198.20 ± 6.50
	MediumGrid	-275.60 ± **2.80**	-260.15 ± 18.20	-315.45 ± 3.10	**-245.80** ± 45.10
	Original	-25.60 ± **2.80**	**108.50** ± 12.40	-45.20 ± 7.50	102.30 ± 8.90
	SmallClassic	-248.40 ± 4.20	-215.80 ± 15.60	-240.15 ± 2.80	**-210.50 ± 2.50**
	Trapped	**-385.40** ± 95.50	-498.20 ± **0.015**	-465.30 ± 12.40	-430.15 ± 6.80
	Tricky	-135.20 ± **0.02**	-110.50 ± 16.80	-160.40 ± 8.50	**-95.40** ± 5.50

centrated between -0.4 and 0.4 without reaching the firing threshold, suggesting increased network sparsity and strong inhibition of overall firing rates.

Figure 11 compares the performance of the baseline PPO algorithm and our proposed OSFD-PPO algorithm under varying observation radius settings. As a classic algorithm in the reinforcement learning domain, PPO typically outperforms baselines such as DQN across numerous tasks, thereby providing a stronger benchmark. Experimental results indicate that in the most severely restricted and challenging environment ($r = 1$), the acute scarcity of extractable effective state information imposes severe exploration difficulties on both algorithms, leading to significant performance variance. Under this extreme setting, the baseline PPO holds a marginal advantage in the final return. However, when the observation radius is moderately expanded to $r = 2$, our proposed method effectively enhances the utilization efficiency of limited observational information, rapidly and stably converging to a high-return regime of approximately 800, whereas ANN-PPO converges much more slowly. Furthermore, in the relatively easier environment with more sufficient information ($r = 3$), OSFD-PPO consistently maintains a faster initial convergence rate and stable asymptotic performance.

As illustrated in Figure 12, we investigate the impact of varying observation radii on the performance of our method. Due to the differing input dimensions of the linear layer across various radii, there are inherent variations in absolute performance. Panels (a) and (b) demonstrate that under smaller observation radii,

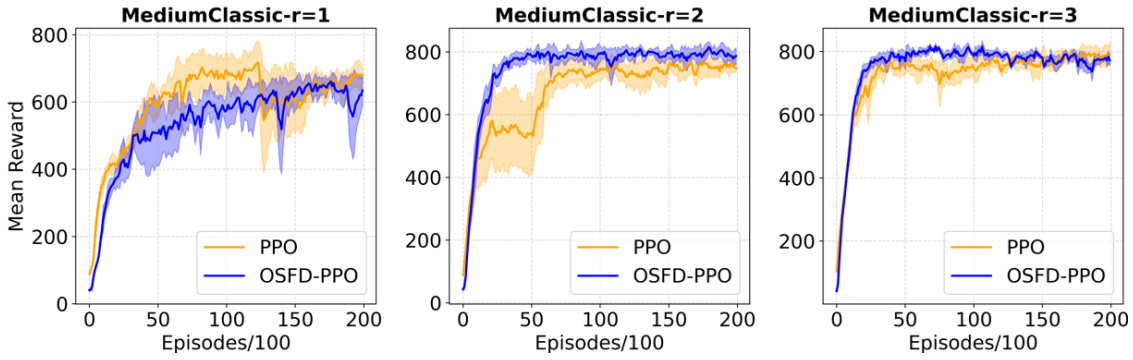

Figure 11: Comparison of Experimental Results for PPO in Pacman Environments with Varying Observation Radii

OSFD exhibits a more pronounced performance advantage. Conversely, as depicted in panel (c), when the observation radius is expanded to 4 grids, this performance gap gradually narrows. Overall, despite the variations in observation ranges, the proposed method consistently maintains a stable advantage and high robustness.

Table 8: Performance comparison of various PPO-based methods across diverse environments, including four tasks from DeepMind Lab.

Domain	**Environment**	**PPO-based Methods**			**SARSA-based Methods**	
		PPO	**SNN-PPO**	**OSFD-PPO**	**SARSA**	**OSFD-SARSA**
Mujoco-Maze	Maze-S-shape	2.374 ± 0.311	2.108 ± 0.470	**2.988 ± 0.012**	0.903 ± 0.033	0.955 ± 0.028
	Maze-U-shape	1.640 ± 0.225	1.685 ± 0.129	**2.031** ± 0.078	0.876 ± **0.026**	0.931 ± 0.063
	Maze-spiral-shape	2.345 ± 0.545	3.137 ± 0.890	**4.013 ± 0.008**	0.821 ± 0.010	0.907 ± 0.015
	Maze-square-random	0.672 ± 0.176	0.546 ± 0.118	**0.972 ± 0.008**	0.084 ± 0.018	0.185 ± 0.015
	Maze-g1 (Success Rate %)	75.2 ± 12.8	68.5 ± 6.3	**94.5 ± 5.1**	43.5 ± 9.2	67.1 ± 8.3
	Maze-g2 (Success Rate %)	62.6 ± 8.46	59.8 ± 10.7	**86.0** ± 7.2	30.4 ± **6.5**	53.0 ± 7.7
DeepMind Lab	Nav-maze-random-goal	23.42 ± 5.83	15.46 ± **1.05**	**25.77** ± 3.75	-	-
	Nav-maze-static	42.81 ± 18.51	37.73 ± 12.94	**51.20 ± 5.91**	-	-
	Seekavoid-arena	**65.65** ± 7.58	58.45 ± **2.06**	64.88 ± 2.21	-	-
	Stairway-to-melon	128.54 ± 21.92	115.69 ± 26.44	**142.24 ± 18.70**	-	-
Pac-Man	MediumClassic ($r = 1$)	643.75 ± 37.12	620.05 ± 22.89	**695.81** ± 35.06	-122.13 ± **7.68**	-12.64 ± 38.16
	MediumClassic ($r = 2$)	745.06 ± 32.76	715.16 ± 25.53	**790.87** ± 5.80	-52.13 ± 161.14	173.07 ± **1.31**
	MediumClassic ($r = 3$)	771.87 ± 18.88	753.89 ± 31.02	**778.94 ± 0.37**	120.05 ± 38.95	272.26 ± 1.28
	Contest	576.88 ± 13.06	558.26 ± **10.08**	**624.75** ± 15.46	-245.61 ± 10.20	-230.11 ± 29.05
	Original	825.41 ± 48.77	796.45 ± 35.99	**934.62** ± 28.65	-63.10 ± **13.78**	55.16 ± 15.05
	Tricky	513.54 ± 21.51	483.65 ± 15.76	**615.05** ± 7.27	-168.22 ± **3.65**	-120.08 ± 6.15

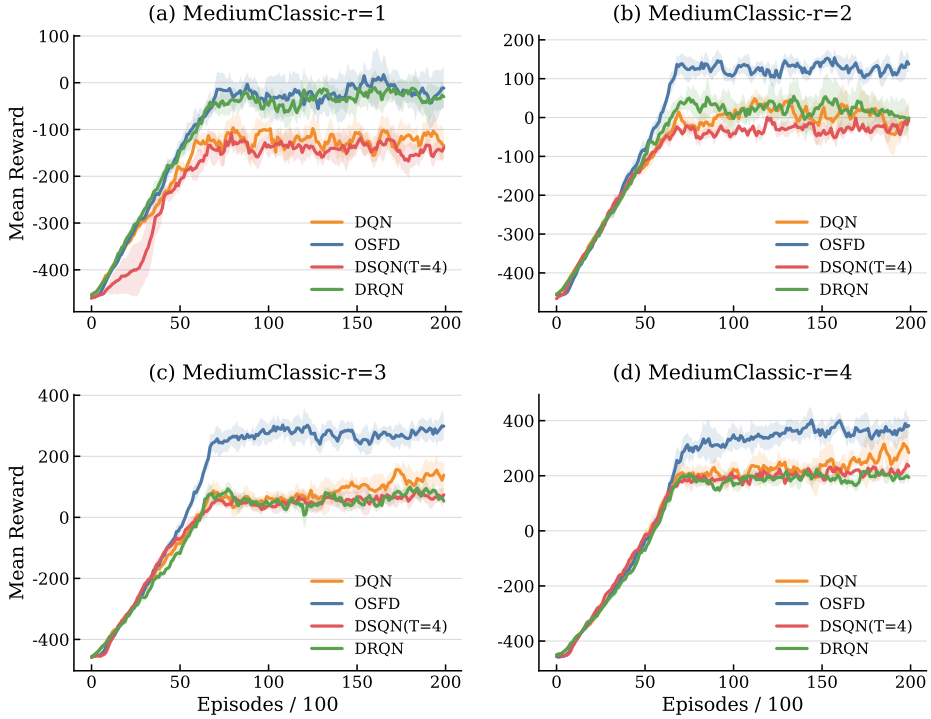

Figure 12: Comparison of the Experimental Performance of OSFD and Baseline Methods Across Different Observation Radii

The global distribution histograms in Figure 13 provide a comprehensive comparison of the statistical characteristics of the evaluated algorithms across multiple core metrics. In the four subplots concerning scores and rewards, the OSFD-DQN algorithm demonstrates a prominent performance advantage: its distributions shift substantially towards the high-return regime on the right, with both the mean score and mean reward decisively penetrating the positive range. In contrast, the samples from baseline algorithms, such as DQN, DRQN, and DSQN, are predominantly concentrated in the zero or negative regions. Regarding survival timesteps, OSFD-DQN exhibits survivability comparable to that of recurrent network-based baselines and significantly outperforms the traditional DQN. Crucially, despite OSFD-DQN achieving the highest actual returns, its mean maximum Q-value remains lower than those of algorithms like DQN and DRQN.

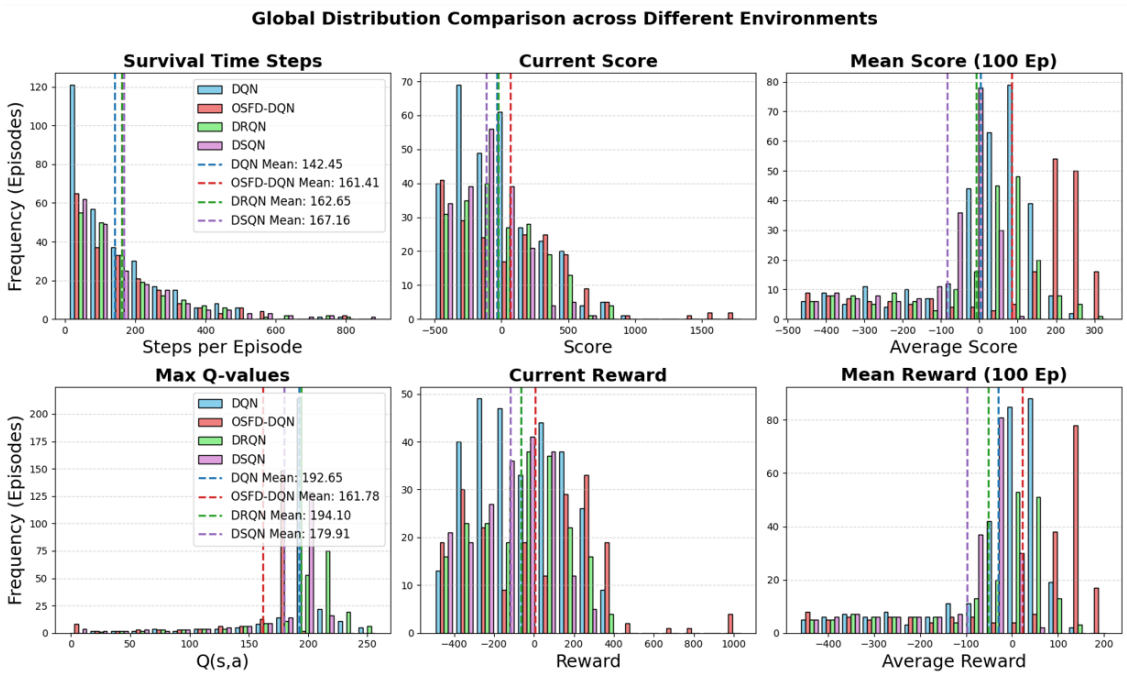

Figure 13: The global distribution histograms provide a comprehensive comparison of the statistical characteristics of the various algorithms across multiple key evaluation metrics.

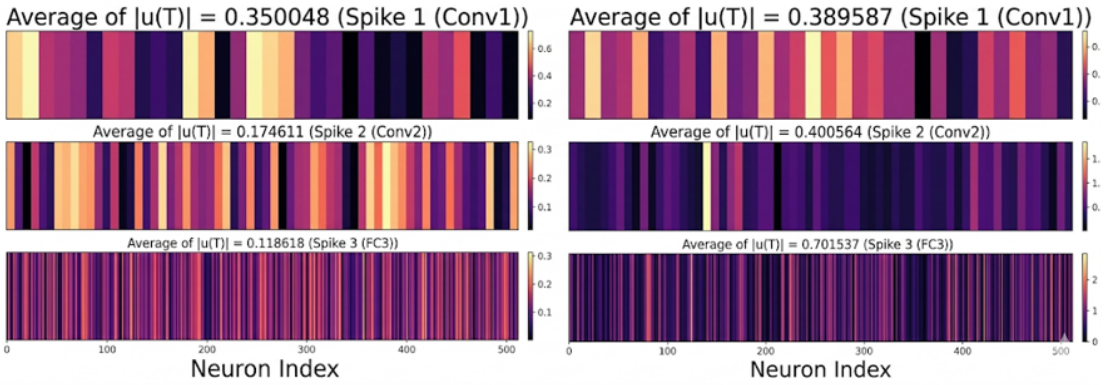

Figure 14: This visualization illustrates the distribution of the absolute membrane potentials $|u(T)|$ of neurons across different layers within the spiking neural network—as employed in the method described in this section—both during the initial stages of reinforcement learning training and following policy convergence.

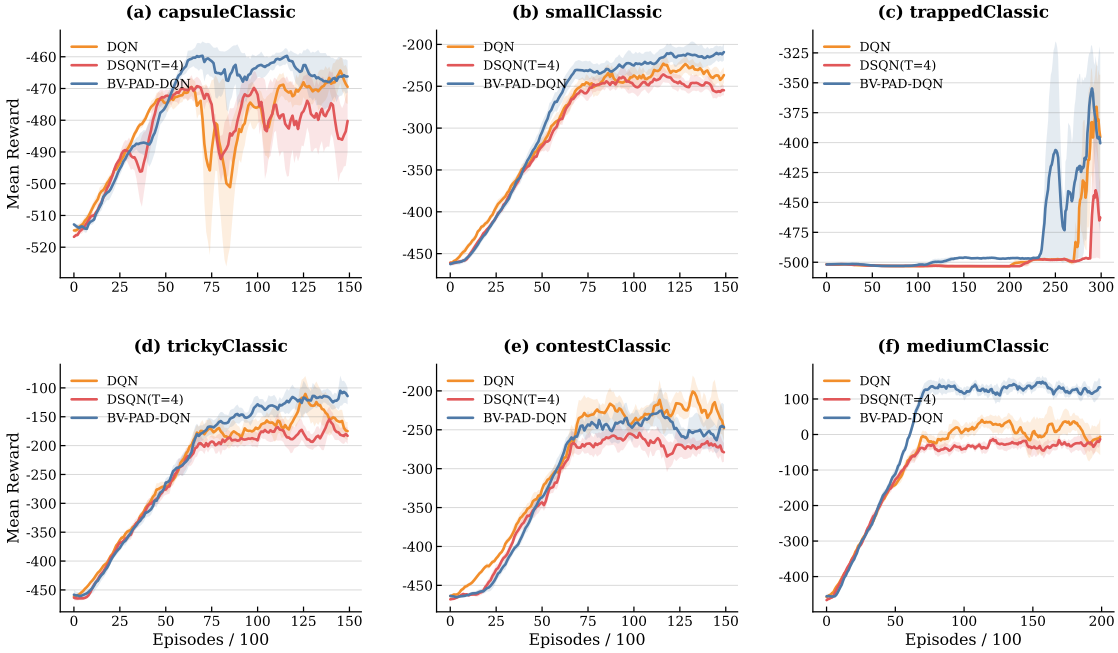

Figure 15: Experimental Results of Pacman Methods in Different Environments

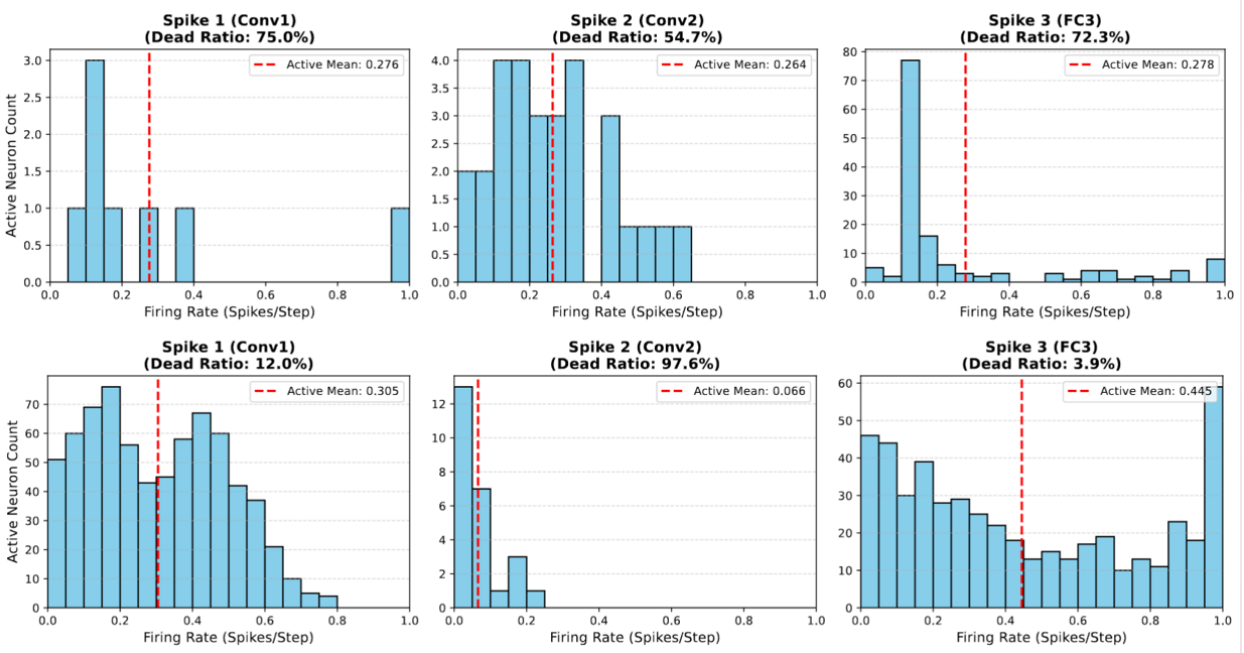

Figure 16: The experiment compares the distribution of neuronal firing rates and the "mortality rate"—defined as the proportion of inactive neurons—across the various layers (Conv1, Conv2, and FC3) of a Spiking Neural Network. This comparison is conducted after the model has been trained for 4,000 iterations in two distinct environments: mediumClassic (shown in the first row) and smallGrid (shown in the second row).