Learning to Walk Randomly

Liyuan Zhang

Abstract

A bipedal robot agent was trained to learn efficient 2D walking using the Soft Actor-Critic (SAC) algorithm. Experiments are conducted in two versions of a Box2D Walker environment: an easy flat terrain and a hardcore terrain with obstacles. In the easy setting, the SAC-trained agent quickly learns a stable walking gait, with episode returns consistently exceeding 300 within roughly 1000 training episodes. In the more challenging hardcore setting, the agent improves its performance and achieves a best episode score of about 301, but its learning curve shows high variance and it does not consistently reach the optimal score. SAC demonstrated strong sample efficiency – the agent converged in a few hundred episodes, but additional strategies are needed for robust performance on very difficult terrains.

1 Methodology

The Soft Actor-Critic (SAC) algorithm [2], an off-policy entropy-regularized actor-critic method, was adopted. SAC optimizes a stochastic policy by augmenting the reward with an entropy term, which encourages exploration and prevents premature convergence. The actor–critic framework consists of a policy (actor) $\pi_\phi(a \mid s)$ and two Q-value function approximators $Q_{\theta_1}(s,a)$ and $Q_{\theta_2}(s,a)$. Off-policy training is enabled by a replay buffer of past transitions for sample-efficient reuse. To stabilize learning, SAC employs twin Q-networks (clipped double Q-learning) [2] to mitigate overestimation, and it uses target network updates with Polyak averaging (soft updates) for the critics.

Each critic Q_{θ_i} is learned by minimizing a mean-squared Bellman error with a clipped double Q target. For a transition (s_t, a_t, r_t, s_{t+1}) sampled from the replay buffer $\mathcal{D}$, the target value is:

$$y_t = r_t + \gamma\Big(\min_{j=1,2} Q_{\bar{\theta}_j}(s_{t+1},\, a'_{t+1}) - \alpha \log \pi_\phi(a'_{t+1} \mid s_{t+1})\Big),$$

where $a'_{t+1} \sim \pi_\phi(\cdot \mid s_{t+1})$ is an action drawn from the current policy and $Q_{\bar{\theta}_j}$ are the delayed target networks. The loss for each critic Q_{θ_i} is then given by Equation (1):

$$J_{Q_i}(\theta_i) \;=\; \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}}\Big[\frac{1}{2}\Big(Q_{\theta_i}(s,a) - \Big(r + \gamma\big(\min_{j=1,2} Q_{\bar{\theta}_j}(s',a') - \alpha \log \pi_\phi(a' \mid s')\big)\Big)\Big)^2\Big] \;, \quad (1)$$

which is minimized to train the critics.

The policy π_ϕ is updated to maximize the expected return *plus* entropy. Equivalently, one can minimize a policy loss J_π that includes an entropy term to encourage exploration. Using the reparameterization trick [2] to backpropagate through sampled actions, the actor objective is given by Equation (2):

$$J_\pi(\phi) \;=\; \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi_\phi(\cdot|s)}\Big[\, \alpha \log \pi_\phi(a \mid s) \;-\; \min_{i=1,2}\, Q_{\theta_i}(s,a)\Big] \;, \quad (2)$$

which corresponds to the negative expected soft Q-value. In practice, using the minimum of the two critic estimates in J_π helps to reduce positive bias in policy gradient estimation.

SAC also introduces an entropy temperature α to balance the entropy term relative to the reward. Rather than treating α as a fixed hyperparameter, SAC adjusts it automatically by minimizing a temperature loss that drives the policy entropy toward a target entropy $\mathcal{H}_{\text{target}}$ [2]. Specifically, α is updated via gradient descent on the loss in Equation (3):

$$J(\alpha) \;=\; \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi_\phi(\cdot|s)}\Big[-\, \alpha \log \pi_\phi(a \mid s) + \mathcal{H}_{\text{target}}\Big] \;, \quad (3)$$

so that $\nabla_\alpha J(\alpha) = 0$ when $\mathbb{E}_{a\sim\pi_\phi(\cdot|s)}\big[-\log \pi_\phi(a \mid s)\big] = \mathcal{H}_{\text{target}}$. In effect, if the policy entropy is lower than desired, α will increase to encourage exploration, and vice versa.

All function approximators (the two critics Q_{θ_i} and the policy π_ϕ) are implemented as multilayer perceptrons (MLPs) with three fully connected hidden layers of 256 units each, using ReLU activations. The stochastic policy outputs Gaussian actions, which are then squashed via tanh to enforce the action bounds of the environment. The SAC agent is trained in an off-policy manner by sampling mini-batches from $\mathcal{D}$ at each gradient step. Additionally, soft target updates are applied to the critic networks: after each gradient step, the target network weights $\bar{\theta}_i$ are slowly moved toward the latest weights θ_i (e.g., $\bar{\theta}_i \leftarrow \tau\,\theta_i + (1-\tau)\,\bar{\theta}_i$ with $\tau \ll 1$). This approach yields stable and sample-efficient learning, allowing the agent to learn to walk efficiently.

To facilitate reproducibility, Table 1 summarizes the key hyperparameters used for SAC training.

Hyperparameter	Value
Learning rate	3×10^{-4}
Discount factor (γ)	0.99
Batch size	256
Replay buffer size	10^6
Soft target update rate (τ)	0.005
Target entropy ($\mathcal{H}_{target}$)	$-\lvert\mathcal{A}\rvert$

Table 1: Key hyperparameters used for SAC training.

2 Convergence Results

2.1 Easy terrain (hardcore=False)

Figure 1 displays the training trajectory under the standard Walker setting. Episode returns (blue crosses) transition from negative values—characteristic of repeated falls during early exploration—to the 300-point threshold within roughly 200–300 episodes. Between 800 and 1000 episodes the running mean (orange) remains consistently above 200, and thereafter the curve plateaus near 320–330. The narrow grey band of one-standard-deviation confirms that performance variability collapses as training progresses, indicating convergence to an essentially optimal gait well before the 1 000-episode mark.

2.2 Hard-core terrain (hardcore=True)

Figure 2 illustrates training under the more demanding Walker variant. Returns trend upward but remain highly erratic. After an extended exploration phase—scores clustered near −100—the running mean first crosses zero at approximately 700 episodes and peaks near 120 by episode 1 200. Isolated episodes achieve returns in the 250–300 range (best score is around 301), yet these spikes are not sustained; subsequent episodes often fall back below 100. The wide grey band throughout reflects persistent high variance, signifying that the learned policy solves only a subset of obstacle configurations and fails to generalise across the entire course within the allotted 1 500 episodes.

2.3 Summary

On flat terrain the SAC agent attains stable returns exceeding 300 in fewer than 1 000 episodes, demonstrating strong sample efficiency and reliable convergence. In contrast, performance in the hard-core setting improves only intermittently; the agent occasionally discovers successful gaits but fails to maintain them, indicating that additional training time or algorithmic refinements are necessary to master the full diversity of obstacles.

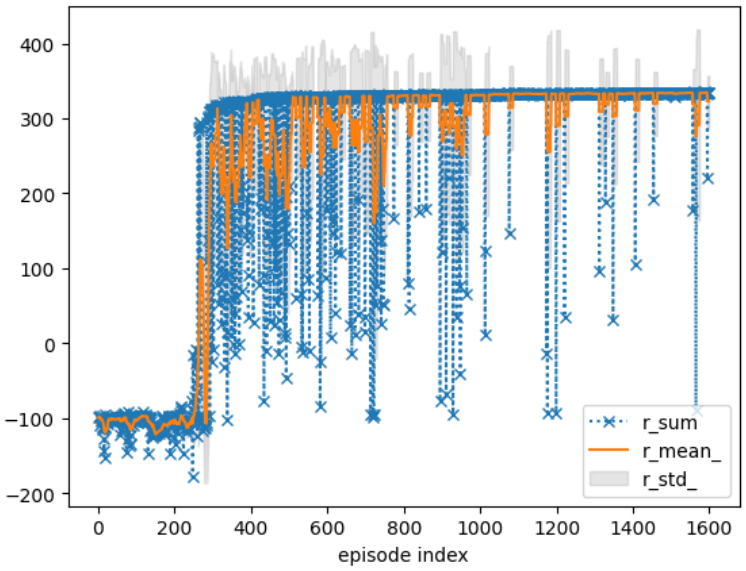

Figure 1: Learning curve for SAC on the easy Walker environment. Blue crosses: episode returns; orange: running mean; grey: ±1 SD.

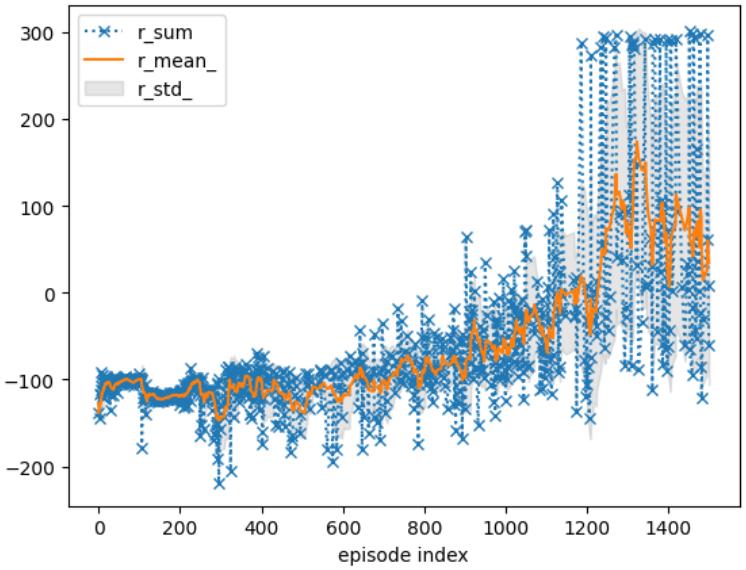

Figure 2: Learning curve for SAC on the hard-core Walker environment. Conventions as in Fig. 1.

3 Limitations

Although SAC attains reliable convergence on the flat terrain, several shortcomings remain. Early-phase instability is noticeable in the easy setting: during roughly the first 300 episodes the return curve exhibits sporadic spikes and dips. These regressions arise from stochastic exploration before the critics have stabilised and subside only after several hundred update cycles.

Performance on the hard-core terrain reveals more critical weaknesses.Returns remain highly volatile for the full 1 500-episode budget; isolated high-scoring episodes are often followed by sharp losses, implying insufficient robustness to the diverse obstacle patterns.The learned policy therefore generalises only partially across the range of terrain configurations.A further practical inconvenience is the considerable wall-clock time required for hard-core training—substantially longer than for the easy variant—because the environment step cost is higher and additional episodes are needed before meaningful improvement appears.

In combination, these observations suggest that the SAC, with the current hyper-parameter schedule, is inadequate for consistently mastering highly irregular walking surfaces within reasonable computational budgets.

Future Work

To address the above limitations and further improve the walking agent, we propose several directions for future work:

3.1 Algorithmic alternatives

Train the bipedal walker with other deep RL algorithms like Twin Delayed DDPG (TD3)[1] or Proximal Policy Optimization (PPO)[3] for comparison. These algorithms could offer different stability or exploration characteristics, and evaluating them would determine if SAC's performance is indeed superior in this domain or if another method yields more robust learning.

3.2 Curriculum learning & extended training

Introduce a curriculum for the hardcore environment or simply allow longer training time. A curriculum could start the agent on easier versions of the terrain and gradually increase difficulty (e.g. progressively larger obstacles), helping the policy to incrementally adapt. Alternatively, training for many more episodes might allow the SAC agent to eventually converge in hardcore mode. This would test the sample efficiency limits and whether the performance plateau seen at 1500 episodes can be overcome with more experience.

3.3 Reward shaping and hierarchical RL

Incorporate learned reward shaping or a hierarchical learning approach to guide the agent in complex terrains. For instance, shaping the reward function to give the agent intermediate feedback (such as bonuses for overcoming obstacles or maintaining balance) could improve learning in the hardcore environment. Similarly, a hierarchical policy (where a high-level controller sets sub-goals or modes for a low-level walking controller) might simplify the task – the high-level policy could handle strategic navigation of obstacles while the low-level policy deals with leg coordination. Such approaches could reduce the burden on a single flat policy learning everything from scratch, leading to more robust walking behavior.

References

[1] Scott Fujimoto, Herke Hoof, and David Meger. "Addressing function approximation error in actor-critic methods". In: *International Conference on Machine Learning*. PMLR. 2018, pp. 1587–1596.

[2] Tuomas Haarnoja et al. “Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor”. In: *ICML*. 2018.

[3] John Schulman et al. “Proximal Policy Optimization Algorithms”. In: *CoRR* abs/1707.06347 (2017). arXiv: 1707.06347. URL: http://arxiv.org/abs/1707.06347.