Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Generative Adversarial Network for Goal Oriented Subgoal Training Exploration and Rewards

William Rudnisky

wrudnis1@binghamton.edu

Binghamton University

Yong Wang

Binghamton University

Research Article

Keywords: Generative AI, Deep Reinforcement Learning, Conditional Generative Adversarial Networks, Intrinsic Reward Shaping

Posted Date: November 5th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11239201/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Generative Adversarial Network for Goal Oriented Subgoal Training Exploration and Rewards

William Rudnisky and Yong Wang

Systems Science and Industrial Engineering, SUNY Binghamton, 4400 Vestal Pkwy E, Binghamton, 13902, New York, USA.

*Corresponding author(s). E-mail(s): yongwang@binghamton.edu;
Contributing authors: wrudnis1@binghamton.edu;

Abstract

Exploration and long-horizon credit assignment remain central challenges in continuous-control reinforcement learning, particularly as environment complexity increases. We propose a Generative Adversarial Network for Goal-oriented Subgoal Training, Exploration, and Rewards (GANGSTER), a generative subgoal framework that augments off-policy reinforcement learning with a conditional generative adversarial network trained using structured loss components capturing subgoal desirability, reachability, and diversity. GANGSTER is integrated with both Twin Delayed Deep Deterministic Policy Gradient (TD3) and Soft Actor Critic (SAC) without modifying their core update rules. Across multiple Multi-Joint dynamics with Contact (MuJoCo) environments, the approach can improve sample efficiency, post-solution stability, and final policy performance relative to baseline algorithms, with the largest gains observed in high-complexity tasks. Empirical analysis shows that performance improvements scale with environment complexity and that individual loss components play interpretable and complementary roles. In Ant-v5, GANGSTER reduces time-to-solve by approximately 31% while increasing final performance by 53% compared to baseline TD3, and for SAC in the HalfCheetah-v5 environment, GANGSTER increases final performance by 10% while improving solution consistency to over 99%.

Keywords: Generative AI, Deep Reinforcement Learning, Conditional Generative Adversarial Networks, Intrinsic Reward Shaping

1 Introduction

Reinforcement learning (RL) has achieved strong performance in continuous-control domains, yet effective exploration and long-horizon credit assignment remain persistent challenges, particularly in high-dimensional environments with sparse or delayed rewards [1–3]. While modern off-policy algorithms such as Soft Actor-Critic (SAC) and Twin Delayed Deep Deterministic Policy Gradient (TD3) provide stable optimization and strong asymptotic performance, they rely largely on undirected exploration mechanisms that scale poorly as task complexity increases. As a result, learning efficiency and reliability often degrade in environments with long horizons, complex dynamics, or large state–action spaces.

Hierarchical reinforcement learning (HRL) has long been proposed as a means to address these limitations by decomposing long-horizon tasks into sequences of intermediate objectives or subgoals [4–6]. However, practical HRL methods frequently require manually specified hierarchies, task-specific abstractions, or auxiliary controllers, limiting their generality and applicability. More recent approaches attempt to automatically discover subgoals from experience, yet many rely on heuristics, fixed clustering strategies, or implicit representations that provide limited interpretability and weak empirical guarantees across environments.

An alternative perspective is to view subgoals as samples from a structured distribution conditioned on the agent's experience. From this viewpoint, effective subgoals should satisfy three intuitive properties: they should correspond to desirable outcomes, be reachable from the agent's current state, and exhibit sufficient diversity to avoid collapse to trivial or redundant targets. Explicitly modeling and balancing these properties offers a principled way to guide exploration without imposing rigid task decompositions or modifying the underlying RL algorithm.

In this work, we propose GANGSTER, a generative subgoal framework that augments off-policy reinforcement learning through a conditional generative adversarial network (cGAN). GANGSTER learns a distribution over subgoals conditioned on the agent's current state and reward context, and generates candidate subgoals that are evaluated using structured loss components capturing desirability, reachability, and diversity. Rather than replacing or restructuring the base RL algorithm, GANGSTER operates as a modular augmentation: generated subgoals define an intrinsic shaping signal that is combined with the environment reward, while the agent continues to learn using standard TD3 or SAC updates.

The proposed approach is designed to be algorithm-agnostic and environment-independent. GANGSTER integrates seamlessly with existing off-policy methods, requires no manually specified hierarchies or task-specific engineering, and introduces interpretable mechanisms for controlling subgoal quality. Importantly, the generative model is trained directly from experience using a structured objective, allowing the influence of individual loss components to be analyzed and ablated.

We evaluate GANGSTER across multiple MuJoCo continuous-control environments using both TD3 and SAC. Our experimental analysis examines sample efficiency, post-solution stability, final performance, and consistency across seeds. In addition, we conduct a detailed ablation study to isolate the contributions of individual loss components and analyze how performance gains scale with environment complexity. The

results show that GANGSTER improves learning speed and stability relative to baseline algorithms in moderate to high-complexity environments, with the largest gains observed in high-complexity tasks, while preserving performance in simpler domains.

The contributions of this work are threefold:

- Introduced **GANGSTER**, a generative subgoal framework that uses structured loss terms to guide exploration in continuous-control reinforcement learning.
- Demonstrated that GANGSTER integrates with standard off-policy algorithms without modifying their core update rules, improving sample efficiency and training stability across multiple environments.
- Provided an empirical analysis showing that the benefits of structured generative subgoals scale with environment complexity, while individual loss components contribute complementary and interpretable effects.

1.1 Motivation

Deep reinforcement learning has achieved strong empirical performance in continuous control tasks. However, learning remains fragile in environments characterized by sparse rewards, delayed credit assignment, and high-dimensional state spaces. In such settings, agents often struggle with inefficient exploration, leading to slow convergence, high variance across seeds, and sensitivity to hyperparameters.

A central challenge lies in the mismatch between the reward signal provided by the environment and the structural information required to guide learning. Standard off-policy methods such as TD3 and SAC optimize value functions using replayed transitions, but exploration is typically driven by action noise or entropy regularization. These mechanisms are local in nature and do not explicitly encode longer-horizon structure. As task complexity increases, purely noise-driven exploration becomes increasingly inefficient.

Hierarchical reinforcement learning approaches attempt to address this issue by introducing temporally extended actions or subgoal abstractions. However, many such methods require architectural modifications, subgoal-conditioned policies, or additional supervisory signals. This can increase implementation complexity and introduce stability challenges.

An alternative perspective is to treat subgoal discovery as a generative modeling problem. Rather than predefining hierarchical structure or conditioning the policy on explicit goals, one can learn to propose intermediate states that are simultaneously desirable in terms of expected return, reachable from the current state, and diverse enough to promote broad state-space coverage. If such subgoals can be generated reliably, they may serve as structured exploration anchors while preserving the underlying off-policy optimization dynamics.

The motivation behind GANGSTER is therefore twofold. First, we aim to introduce structure into exploration without modifying the core update rules of strong off-policy baselines. Second, we seek a mechanism whose benefit scales with environment complexity, rather than diminishing as tasks become more difficult.

By framing subgoal generation as a conditional generative modeling problem and integrating it through reward shaping rather than architectural redesign, we retain

the stability properties of TD3 and SAC while augmenting their exploratory behavior. This approach provides a modular pathway for structured exploration in continuous control domains where sparse or delayed rewards otherwise hinder efficient learning.

1.2 Contributions

A central question motivating this work is whether *generative subgoal discovery can improve exploration efficiency and policy stability in increasingly complex continuous-control environments*. While prior approaches to exploration rely on stochasticity, intrinsic motivation, or hierarchical policy structures, they often lack explicit control over the quality and structure of intermediate objectives. This work addresses that gap by introducing a generative framework for structured subgoal discovery and integration into off-policy reinforcement learning.

The primary contributions of this work are as follows:

- **A generative subgoal framework for continuous control** We introduce GANGSTER, a conditional generative adversarial framework that learns a distribution over subgoals conditioned on the agent's state and reward context. Unlike prior approaches that rely on hierarchical policies or goal relabeling techniques such as Hindsight Experience Replay (HER) [7], the proposed method explicitly generates candidate subgoals, enabling structured exploration beyond previously observed trajectories.
- **Structured subgoal optimization via interpretable constraints** Subgoal generation is formulated as a multi-objective optimization problem incorporating three interpretable components: *desirability*, *reachability*, and *diversity*. Desirability encourages alignment with high-value states under the learned critic, reachability promotes feasibility within the agent's current state distribution, and diversity prevents collapse to redundant or trivial goals. This explicit decomposition contrasts with intrinsic motivation methods such as curiosity-driven exploration [8] and random network distillation [9], which provide undirected exploration signals without structured control over intermediate objectives.
- **Algorithm-agnostic integration with off-policy reinforcement learning** The proposed framework augments standard off-policy algorithms, specifically TD3 [2] and SAC [3], without modifying their core update rules. GANGSTER operates by shaping the reward signal and influencing the data distribution in the replay buffer, preserving the stability and convergence properties of the underlying actor–critic methods while enhancing exploration.
- **Systematic empirical evaluation across continuous-control benchmarks** We evaluate the proposed framework on a suite of MuJoCo environments [10] spanning a range of control complexities, from low-dimensional stabilization tasks to high-dimensional locomotion. Performance is assessed using multiple metrics capturing learning efficiency, stability, and final policy quality, providing a comprehensive evaluation beyond standard reward curves.
- **Complexity-dependent analysis of subgoal-driven learning** We demonstrate that the benefits of structured generative subgoals scale with environment complexity. While performance gains are modest in simple tasks, substantial improvements

are observed in high-dimensional and unstable environments, where traditional exploration strategies are insufficient. This analysis provides insight into when generative subgoal discovery is most effective and highlights its role as a targeted solution for exploration-limited regimes.

2 Background

2.1 Off-Policy Reinforcement Learning

Off-policy reinforcement learning methods optimize a target policy using data collected under a potentially different behavior policy. This decoupling enables improved sample efficiency by reusing past experience stored in a replay buffer. In continuous control settings, off-policy algorithms have demonstrated strong performance due to their ability to leverage large replay datasets and stabilize training through target networks and delayed updates.

In actor–critic formulations, a parametric policy $\pi_\phi(a \mid s)$ (actor) is optimized using value function approximations $Q_\theta(s, a)$ (critics). Transitions (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer $\mathcal{D}$ and sampled uniformly (or via prioritized schemes) to compute temporal-difference targets.

Let y_t denote the target value:

$$y_t = r_t + \gamma \, \mathbb{E}_{a' \sim \pi} \left[Q_{\theta'}(s_{t+1}, a') \right], \tag{1}$$

where θ' denotes target network parameters. The critic is trained by minimizing the Bellman residual:

$$\mathcal{L}_{critic} = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} \left[\left(Q_\theta(s, a) - y_t \right)^2 \right]. \tag{2}$$

The actor update depends on the specific algorithmic formulation. In this work, we integrate GANGSTER with two widely adopted off-policy continuous-control algorithms: TD3 and SAC.

2.1.1 Twin Delayed Deep Deterministic Policy Gradient (TD3)

TD3 is a deterministic actor–critic algorithm designed to mitigate overestimation bias in Q-learning for continuous action spaces. It extends Deep Deterministic Policy Gradient (DDPG) by introducing three key modifications: clipped double Q-learning, delayed policy updates, and target policy smoothing.

Clipped Double Q-Learning

TD3 maintains two independent critics, Q_{θ_1} and Q_{θ_2}. The target value uses the minimum of the two estimates:

$$y_t = r_t + \gamma \min_{i=1,2} Q_{\theta_i'} \left(s_{t+1}, \pi_{\phi'}(s_{t+1}) \right), \tag{3}$$

which reduces positive bias in value estimation.

Target Policy Smoothing

To improve robustness, noise is added to the target action:

$$a' = \pi_{\phi'}(s_{t+1}) + \epsilon, \quad \epsilon \sim \text{clip}\left(\mathcal{N}(0, \sigma^2), -c, c\right). \tag{4}$$

The smoothed action is used inside the target critic.

Delayed Policy Updates

The actor is updated less frequently than the critics. Every d critic updates, the policy is optimized via:

$$\nabla_\phi J(\phi) = \mathbb{E}_{s\sim\mathcal{D}}\left[\nabla_a Q_{\theta_1}(s,a)\big|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)\right]. \tag{5}$$

This delayed update stabilizes learning by allowing critics to converge before updating the actor.

In our implementation, TD3 serves as the base optimizer. GANGSTER modifies only the reward signal and exploration structure; TD3 update rules remain unchanged.

2.1.2 Soft Actor–Critic (SAC)

SAC is a stochastic off-policy actor–critic algorithm that maximizes both expected return and policy entropy. By incorporating entropy regularization, SAC encourages exploration and improves stability across diverse tasks.

The SAC objective augments the standard return with an entropy term weighted by temperature parameter α:

$$J(\pi) = \sum_t \mathbb{E}_{(s_t,a_t)\sim\pi}\left[r_t + \alpha\mathcal{H}(\pi(\cdot \mid s_t))\right]. \tag{6}$$

Soft Q-Function Update

SAC maintains two critics and uses the minimum for stability. The target value incorporates the entropy term:

$$y_t = r_t + \gamma\left(\min_{i=1,2} Q_{\theta_i'}(s_{t+1}, a') - \alpha \log \pi_\phi(a' \mid s_{t+1})\right), \tag{7}$$

where $a' \sim \pi_\phi(\cdot \mid s_{t+1})$.

Policy Update

The policy is updated by minimizing:

$$\mathcal{L}_{actor} = \mathbb{E}_{s\sim\mathcal{D}, a\sim\pi_\phi}\left[\alpha \log \pi_\phi(a \mid s) - Q_\theta(s,a)\right]. \tag{8}$$

In practice, the temperature parameter α may be fixed or automatically tuned to match a target entropy level.

Within GANGSTER, SAC is integrated analogously to TD3: intrinsic reward augments the environment reward, while actor and critic optimization remain unchanged.

3 Related Work

3.1 Hierarchical Reinforcement Learning and Subgoal Discovery

Hierarchical reinforcement learning (HRL) has long been proposed as a framework for addressing long-horizon decision-making by decomposing complex tasks into sequences of intermediate objectives or temporally extended actions. Early approaches such as the options framework and its extensions introduced the concept of reusable skills or sub-policies that operate over multiple time steps, enabling temporal abstraction and improved learning efficiency. The Option-Critic architecture [11] formalizes this idea by jointly learning intra-option policies and termination conditions, allowing hierarchical structure to emerge from experience.

Subsequent work has explored more explicit subgoal-based formulations of hierarchy. FeUdal Networks [12] introduce a manager–worker architecture in which a high-level controller proposes goals in a learned embedding space, while a low-level policy executes actions to achieve them. Similarly, HIRO [6] and Hierarchical Actor-Critic (HAC) [13] extend hierarchical learning to continuous control by learning goal-conditioned policies across multiple temporal scales. These approaches demonstrate that structured subgoal decomposition can significantly improve performance in long-horizon tasks. Recent methods have also explored imagined intermediate states defined through learned reachability estimates [14], latent subgoal spaces learned from slow dynamics [15], and stable subgoal representations coupled with active hierarchical exploration [16]. More recent work has improved subgoal quality and temporal abstraction through landmark-guided generation [17], timed subgoals for dynamic environments [18], and graph-based decoupling of high- and low-level horizons [19]. Recent hierarchical reinforcement learning methods have also incorporated latent landmark representations and structured subgoal selection strategies to better balance exploration and exploitation in sparse-reward environments [20].

Despite their effectiveness, HRL methods often require architectural modifications, additional policy levels, or carefully designed training procedures. These requirements can introduce instability and increase implementation complexity, particularly in continuous control settings. Moreover, subgoal representations are frequently learned implicitly, limiting interpretability and control over the generated objectives. Related limitations have been addressed through fast and slow curiosity signals that combine local and long-horizon exploration [21], as well as hierarchical learning that uses high-level human preferences, curiosity, and automatic subgoal discovery [22]. Recent work on intrinsic reward shaping has further shown that adaptive intrinsic signals can improve exploration efficiency in deep reinforcement learning [23]. Building on this direction, [24] explored probabilistic subgoal representations for hierarchical reinforcement learning, demonstrating that structured latent subgoal modeling can improve robustness and transferability across stochastic environments. Recent work on subgoal generation has emphasized the importance of conditioning generated subgoals on task context and estimated value in order to balance exploration and exploitation during policy learning [25].

In contrast, the approach proposed in this work avoids explicit hierarchical policy design. Rather than introducing additional controllers, GANGSTER generates subgoals externally using a generative model and integrates them through reward shaping. This preserves the simplicity and stability of standard off-policy methods while providing a mechanism for structured exploration.

3.2 Exploration in Continuous Control

Efficient exploration remains a central challenge in reinforcement learning, particularly in environments characterized by sparse rewards or high-dimensional state spaces. Traditional approaches rely on stochastic policies or action noise, as seen in algorithms such as Deep Deterministic Policy Gradient (DDPG) [26], Twin Delayed Deep Deterministic Policy Gradient (TD3) [2], and Soft Actor-Critic (SAC) [3]. While these methods provide strong optimization performance, their exploration strategies are largely local and may scale poorly with task complexity.

To address this limitation, a wide range of intrinsic motivation methods have been proposed. Curiosity-driven exploration approaches, such as the Intrinsic Curiosity Module (ICM) [8], reward agents for predicting the consequences of their actions, encouraging exploration of novel or unpredictable states. Random Network Distillation (RND) [9] similarly incentivizes novelty by measuring prediction error relative to a fixed random network. Variational information-based methods such as VIME [27] quantify uncertainty in the agent's dynamics model to guide exploration, while count-based approaches approximate state visitation frequencies to encourage coverage of the state space [28]. Noisy Networks [29] incorporate parameter noise directly into the policy to induce exploration. Recent goal-conditioned reinforcement learning approaches have shown that entropy-driven subgoal exploration can improve exploration efficiency and long-horizon policy learning in sparse-reward environments [30].

Although these methods have demonstrated success, they generally provide undirected or novelty-based exploration signals that are not explicitly aligned with task objectives. As a result, exploration may be inefficient in environments where meaningful progress requires coordinated transitions through specific regions of the state space.

An alternative direction involves goal-conditioned reinforcement learning, where policies are trained to achieve arbitrary target states. Universal Value Function Approximators (UVFA) [31] extend value functions to incorporate goal representations, enabling generalization across multiple objectives. Hindsight Experience Replay (HER) [7] further improves sample efficiency by relabeling failed trajectories with alternative goals, allowing agents to learn from unsuccessful experiences.

Recent goal-generation methods have begun to move beyond uniform or replay-buffer-based goal sampling. Planning Exploratory Goals directly optimizes goal commands for their expected exploration value [32], while Stein Variational Goal Generation learns a diverse distribution of goals with intermediate difficulty based on the agent's current goal-reaching capability [33]. These methods demonstrate the value of adapting goal selection to both exploration potential and agent competence, although they do not use an adversarial generator with explicit desirability, reachability, and diversity terms.

While many goal-conditioned approaches derive goals from observed trajectories or learned world models, GANGSTER explicitly models a generative distribution over subgoals. This enables exploration beyond previously observed experience while maintaining alignment with value estimates and feasibility constraints.

3.3 Generative Models and Adversarial Approaches in Reinforcement Learning

3.3.1 Generative Models in Reinforcement Learning

Generative modeling has emerged as a powerful tool for representation learning and data generation, with Generative Adversarial Networks (GANs) [34] providing a flexible framework for learning complex data distributions. Conditional GANs (cGANs) [35] extend this approach by conditioning generation on auxiliary inputs, enabling controlled synthesis of samples based on contextual information. Variants such as Wasserstein GANs (WGAN) [36] improve training stability and convergence properties, while InfoGAN [37] introduces latent structure to enhance interpretability.

In reinforcement learning, generative models have been applied to a variety of tasks, including representation learning, model-based prediction, and data augmentation. For example, energy-based policy formulations [38] and large-scale robotic learning systems [39] demonstrate how learned distributions can guide decision-making in complex environments. However, the use of generative models for explicit subgoal discovery remains relatively underexplored. World-model-based approaches have also demonstrated that learned generative representations can improve exploration and planning efficiency in sparse-reward goal-conditioned reinforcement learning tasks [40].

Most existing approaches either rely on implicit latent representations or use generative models to approximate environment dynamics rather than directly generating actionable intermediate objectives. As a result, they do not provide explicit control over the structure or quality of subgoals used to guide exploration. Recent work has explored the use of bidirectional generative adversarial networks (BiGANs) for novelty-driven exploration, where reconstruction and feature-matching errors are leveraged to estimate state novelty and provide intrinsic reward signals [41].

The proposed GANGSTER framework differs from prior work by directly modeling a conditional distribution over subgoals using a generative adversarial approach. By incorporating structured loss components corresponding to desirability, reachability, and diversity, the framework provides explicit and interpretable control over the generated objectives. This allows the generative model to produce subgoals that are not only novel but also aligned with the agent's value function and feasible within the environment dynamics.

3.3.2 Adversarial and Goal-Generation Approaches

The integration of generative modeling with reinforcement learning has been explored across several domains, including exploration, imitation learning, policy modeling, and data augmentation. These approaches demonstrate that generative models can provide structured signals to improve learning efficiency, but they differ significantly in how generation is used within the reinforcement learning pipeline.

One line of work incorporates generative adversarial networks into exploration mechanisms. Generative Adversarial Exploration (GAEX) [42] introduces a GAN-based intrinsic reward signal, where the discriminator identifies underexplored regions of the state space and encourages the agent to visit them. This approach highlights the potential of adversarial training to guide exploration; however, the generated samples are used primarily to define novelty signals rather than explicit intermediate objectives.

Another class of methods uses reinforcement learning to control or optimize generative models. RL-GAN-Net [43] employs a reinforcement learning agent to control the latent input of a GAN for point cloud generation, effectively treating the generative model as an environment to be optimized. Similarly, SeqGAN [44] formulates sequence generation as a reinforcement learning problem, where the generator is treated as a stochastic policy and trained using policy gradient methods with feedback from a discriminator. These approaches demonstrate the bidirectional relationship between generative modeling and reinforcement learning, but they focus on improving generative outputs rather than improving RL exploration or control policies.

Generative models have also been used for data augmentation and representation learning in reinforcement learning. Recent work explores generating synthetic states or transitions to improve sample efficiency and exploration coverage [45]. Such approaches increase diversity in training data but typically operate at the level of replay buffer augmentation rather than explicitly modeling structured intermediate objectives.

In addition, generative modeling has been integrated into model-based reinforcement learning, where learned generative models approximate environment dynamics or provide sampling oracles for planning [46]. These methods focus on improving planning efficiency and theoretical guarantees but do not directly address structured subgoal generation as a mechanism for guiding exploration.

More broadly, recent surveys highlight that generative models in reinforcement learning are commonly used for three purposes: (i) reward shaping or signal generation, (ii) state representation learning, and (iii) planning and exploration support [47]. However, most existing approaches either rely on implicit latent representations or generate signals that are not explicitly interpretable in terms of task-relevant intermediate objectives.

In contrast, GANGSTER differs from prior work in several key ways. First, it directly models a conditional distribution over subgoals rather than generating auxiliary signals or synthetic data. Second, subgoal generation is governed by explicit and interpretable constraints—desirability, reachability, and diversity—providing structured control over the properties of generated objectives. Finally, the generated subgoals are integrated into reinforcement learning through intrinsic reward shaping without modifying the underlying policy or value function architectures.

This distinguishes GANGSTER from prior approaches that rely on heuristic goal selection, implicit latent structure, or indirect generative signals, positioning it as a unified framework for structured subgoal discovery in continuous-control reinforcement learning.

3.4 Summary

In summary, prior work on hierarchical reinforcement learning, intrinsic motivation, and goal-conditioned policies has provided important insights into improving exploration and learning efficiency. However, these approaches either require architectural modifications, rely on implicit representations, or generate exploration signals that are not explicitly aligned with task objectives.

GANGSTER builds on these foundations by introducing a unified framework for generative subgoal discovery that combines the strengths of these approaches while addressing their limitations. By integrating a conditional generative model with structured constraints into standard off-policy reinforcement learning, the proposed method enables scalable and interpretable exploration in complex continuous-control environments.

4 Methodology

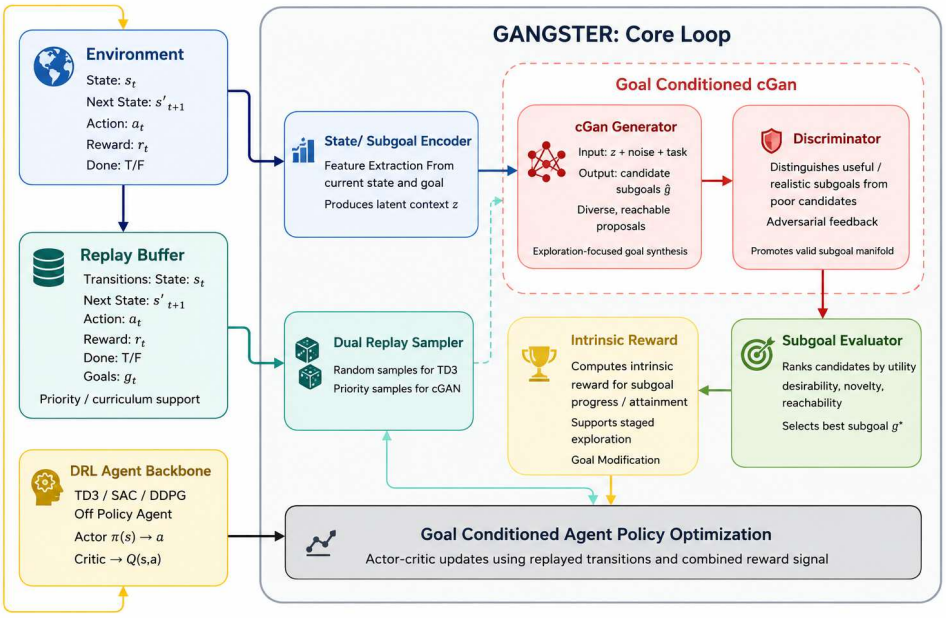


Fig. 1 GANGSTER Framework Architecture

This section describes the GANGSTER framework and its integration with off-policy reinforcement learning algorithms. GANGSTER is designed as a modular augmentation layer that introduces structured subgoal generation and intrinsic reward shaping without modifying the underlying actor–critic update rules.

Our approach consists of three primary components: (i) a conditional generative model that proposes candidate subgoals, (ii) a structured scoring mechanism that selects a feasible and desirable subgoal, and (iii) an intrinsic reward formulation that guides exploration toward the selected subgoal while preserving standard off-policy optimization.

Importantly, GANGSTER does *not* condition the actor or critic networks on subgoals. Instead, it modifies only the reward signal through intrinsic shaping and dynamic reselection of intermediate targets. This design preserves the stability properties of strong baselines such as TD3 and SAC while augmenting exploration with learned structure.

At a high level, the framework operates as follows. At the beginning of an episode (or when a subgoal is achieved), the generator produces a set of candidate subgoals conditioned on the current state. Each candidate is scored using a linear combination of predicted desirability and reachability. The highest-scoring subgoal is selected and used to compute intrinsic reward at each subsequent transition. The augmented reward is stored in the replay buffer and used to update the base off-policy agent via its standard optimization rules.

Algorithm 1 GANGSTER with Off-Policy RL (TD3/SAC)

Input: base off-policy agent (TD3 or SAC) with policy π
Input: replay buffer $\mathcal{D}$
Input: generator G_θ, discriminator D_ψ
Hyperparameters: β (intrinsic weight), ϵ (subgoal tolerance), K (candidates), w_d, w_r (scoring weights)
Initialize agent networks and target networks
Initialize G_θ and D_ψ
Initialize replay buffer $\mathcal{D} \leftarrow \emptyset$
for episode $= 1$ to M **do**
    Receive initial state s_0
    Sample $z \sim \mathcal{N}(0, I)$
    Generate candidates $\{(\tilde{g}_i, \hat{r}_i)\}_{i=1}^{K} \leftarrow G_\theta(s_0, z)$
    Compute scores $S_i \leftarrow w_d \hat{r}_i + w_r S_{\text{reach}}(\tilde{g}_i; s_0)$ for $i = 1, \ldots, K$
    Select $g^* \leftarrow \arg\max_{\tilde{g}_i} S(\tilde{g}_i; s_0)$
    for $t = 0$ to $T - 1$ **do**
        Select action $a_t \sim \pi(\cdot \mid s_t)$
        Execute a_t, observe (r_t, s_{t+1}, d_t) ▷ $d_t \in \{0, 1\}$ termination flag
        Compute $r_{\text{intr}} \leftarrow r_{\text{intr}}(s_{t+1}, g^*)$
        Form $r'_t \leftarrow r_t + \beta\, r_{\text{intr}}$
        Store transition $(s_t, a_t, r'_t, s_{t+1}, d_t)$ in $\mathcal{D}$
        if $\|s_{t+1} - g^*\|_2 \leq \epsilon$ **then**
            Sample $z \sim \mathcal{N}(0, I)$
            Generate $\{(\tilde{g}_i, \hat{r}_i)\}_{i=1}^{K} \leftarrow G_\theta(s_{t+1}, z)$
            Compute $S_i \leftarrow w_d \hat{r}_i + w_r S_{\text{reach}}(\tilde{g}_i; s_{t+1})$ for $i = 1, \ldots, K$
            Select $g^* \leftarrow \arg\max_{\tilde{g}_i} S(\tilde{g}_i; s_{t+1})$
        end if
        if $d_t = 1$ **then**
            break
        end if
        if $|\mathcal{D}|$ is sufficient for training **then**
            Sample minibatch $\mathcal{B} \subset \mathcal{D}$
            Update D_ψ using adversarial loss
            Update G_θ using adversarial + structured losses
            Update base agent (TD3/SAC) on $\mathcal{B}$ using reward r'
        end if
    end for
end for

4.1 Framework Overview

GANGSTER augments standard off-policy reinforcement learning with a structured generative subgoal module. The framework operates alongside a base algorithm (TD3 or SAC) without modifying its core update rules.

At a high level, training proceeds as follows:

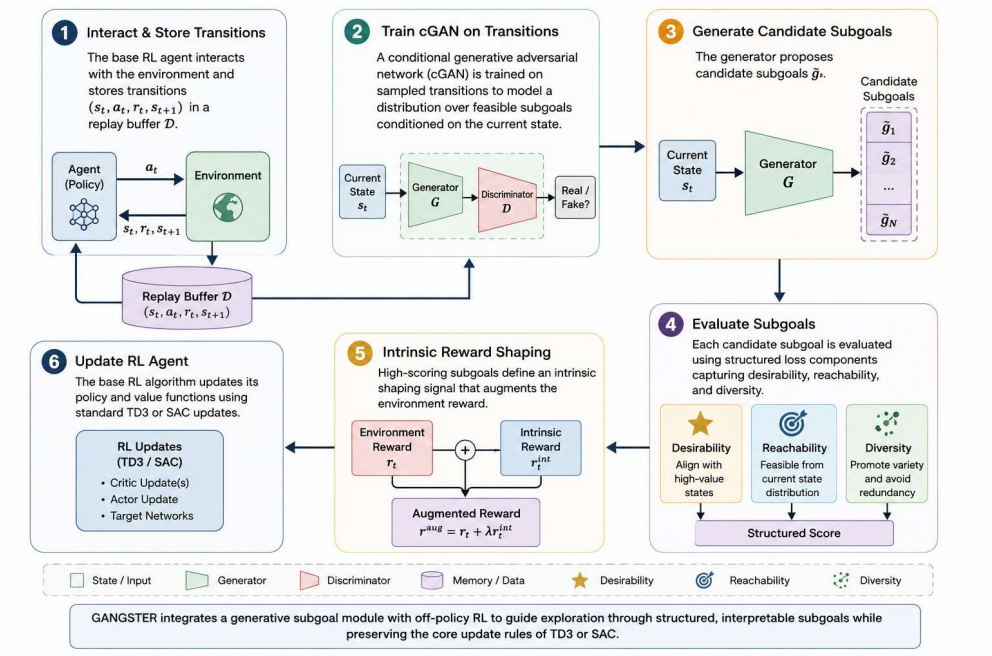


Fig. 2 GANGSTER Training Framework

Importantly, GANGSTER does not alter the actor–critic update equations. It modifies only the data distribution and reward shaping through structured subgoal generation.

4.2 Subgoal Generator (cGAN)

We model subgoal generation using a conditional generative adversarial network. The generator G_θ maps a state and noise vector to a candidate subgoal:

$$\tilde{g} = G_\theta(s, z), \quad z \sim \mathcal{N}(0, I) \tag{9}$$

The discriminator D_ϕ distinguishes between observed transitions and generated subgoal transitions:

$$D_\phi(s, g) \in [0, 1] \tag{10}$$

The adversarial objective is given by:

$$\begin{aligned} \mathcal{L}_{GAN} = \mathbb{E}_{(s,g)\sim\mathcal{D}}[\log D_\phi(s, g)] \\ + \mathbb{E}_{s\sim\mathcal{D}, z\sim\mathcal{N}}[\log(1 - D_\phi(s, G_\theta(s, z)))] \end{aligned} \tag{11}$$

The discriminator is trained to maximize $\mathcal{L}_{GAN}$, while the generator minimizes it.

4.3 Structured Loss Function

To ensure that generated subgoals are useful for exploration, we augment the generator objective with three structured components.

The total generator loss is:

$$\mathcal{L}_G = \mathcal{L}_{GAN} + \lambda_d \mathcal{L}_{des} + \lambda_r \mathcal{L}_{reach} + \lambda_v \mathcal{L}_{div} \tag{12}$$

where $\lambda_d, \lambda_r, \lambda_v$ are weighting coefficients.

4.3.1 Desirability

The desirability term encourages subgoals associated with high expected return. Let $Q(s, g)$ denote the critic-estimated value of reaching subgoal g from state s. We define:

$$\mathcal{L}_{des} = -\mathbb{E}_{s,z}[Q(s, G_\theta(s, z))] \tag{13}$$

This term biases generation toward subgoals predicted to improve long-term return.

4.3.2 Reachability

To prevent unrealistic or excessively distant subgoals, we penalize large state deviations. Let $d(\cdot, \cdot)$ denote a distance metric in state space. We define:

$$\mathcal{L}_{reach} = \mathbb{E}_{s,z}\left[d\left(s, G_\theta(s, z)\right)\right] \tag{14}$$

This term promotes feasible intermediate objectives that stabilize learning.

4.3.3 Diversity

To avoid collapse to trivial or repeated subgoals, we promote diversity within a minibatch. For a batch $\{\tilde{g}_i\}_{i=1}^N$, we define:

$$\mathcal{L}_{div} = -\frac{1}{N(N-1)} \sum_{i \neq j} d(\tilde{g}_i, \tilde{g}_j) \tag{15}$$

Maximizing pairwise dispersion encourages coverage of the state space.

4.4 Subgoal Ranking and Selection

Given a set of candidate subgoals $\{\tilde{g}_i\}_{i=1}^K$ generated by the conditional generator, GANGSTER selects a single subgoal based on a structured scoring function that balances predicted utility and feasibility.

Each candidate subgoal is assigned a score:

$$S(\tilde{g}_i; s) = w_d \hat{r}_i + w_r S_{\text{reach}}(\tilde{g}_i, s) \tag{16}$$

where $\hat{r}_i$ denotes the predicted reward (desirability) associated with the subgoal, and $S_{\text{reach}}(\tilde{g}_i, s)$ is a reachability measure reflecting the distance between the current state s and the candidate subgoal.

The selected subgoal is given by:

$$g^* = \arg\max_{\tilde{g}_i} S(\tilde{g}_i; s) \tag{17}$$

This ranking mechanism ensures that selected subgoals are both aligned with high-value regions of the state space and feasible for the agent to achieve from its current state. By explicitly balancing desirability and reachability, the framework avoids selecting unrealistic or low-value targets while maintaining structured exploration.

4.5 Integration with Off-Policy RL

Generated subgoals define an intrinsic shaping signal:

$$r'_t = r_t + \beta \cdot r_{intrinsic}(s_t, \tilde{g}) \tag{18}$$

where $r_{intrinsic}$ measures progress toward the selected subgoal and β controls scaling.

The base RL algorithm (TD3 or SAC) updates policy and value networks using r'_t without altering its optimization procedure, where

$$r_{\text{intrinsic}}(s_t, \tilde{g}) = d(f(s_t), \tilde{g}) - d(f(s_{t+1}), \tilde{g}) \tag{19}$$

5 Experiment Setup

This section describes the experimental configuration used to evaluate the proposed GANGSTER framework. The goal of the evaluation is to compare the learning efficiency, stability, and final performance of GANGSTER-enhanced agents against standard reinforcement learning baselines under consistent training conditions. All algorithms are evaluated on a set of continuous control benchmarks using identical training budgets, reward structures, and evaluation protocols. The following subsections detail the environments used in the study, the baseline algorithms selected for comparison, the performance metrics used for evaluation, and the statistical methodology used to analyze training results.

The experimental setup utilizes continuous control environments from Gymnasium, with all models implemented in PyTorch and trained using fixed random seeds to ensure reproducibility. A shared replay buffer with a capacity of 100,000 transitions is employed across all methods, incorporating a dual-sampling strategy in which uniform sampling is used for policy learning and prioritized sampling is used for training the generative model based on normalized reward magnitudes.

The proposed GANGSTER framework augments standard off-policy reinforcement learning (TD3 and SAC) with a conditional generative adversarial network (cGAN) that produces candidate subgoals conditioned on the current state, extrinsic reward, and Gaussian noise. At each decision step, $K = 100$ candidate subgoals are generated and ranked using a weighted combination of predicted reward (desirability) and state-distance (reachability), with the highest-scoring subgoal selected. The generator is trained using adversarial loss together with desirability, diversity, and reachability regularization terms, while the discriminator distinguishes between real next-state

transitions and generated subgoals. The reinforcement learning agents (TD3 and SAC) are trained using standard update rules with no architectural modifications, and intrinsic reward shaping is optionally incorporated through a distance-based signal between the current state and the selected subgoal.

Table 1 Hyperparameter Configuration for GANGSTER, TD3, and SAC

Parameter	GANGSTER (cGAN)	TD3	SAC
Replay Buffer Size	100,000	100,000	100,000
Batch Size (RL)	–	256	128
Batch Size (cGAN)	512	–	–
Learning Rate	1×10^{-4}	3×10^{-4}	3×10^{-4}
Discount Factor γ	–	0.99	0.99
Target Update τ	–	0.005	0.005
Policy Noise	–	0.2	–
Noise Clip	–	0.5	–
Policy Update Frequency	–	2	–
Entropy Temperature α	–	–	0.2
Hidden Layer Size	256	400/300	256/128
Noise Dimension	16	–	–
Candidate Subgoals (K)	100	–	–
cGAN Update Interval	Every episode	–	–
Desirability Weight λ_d	Tuned	–	–
Diversity Weight λ_{div}	Tuned	–	–
Reachability Weight λ_r	Tuned	–	–
Intrinsic Reward Weight β	0.1-0.3	–	–

Hyperparameter values for the cGAN components, including the number of candidate subgoals ($K = 100$) and loss weighting coefficients, were selected based on preliminary ablation studies to balance performance, stability, and computational overhead.

5.1 Environments

The proposed GANGSTER framework was evaluated on a set of standard continuous-control reinforcement learning environments from the MuJoCo benchmark suite available through Gymnasium [48]. These environments are widely used for evaluating deep reinforcement learning algorithms due to their varying levels of dynamical complexity, dimensionality, and control difficulty.

The selected tasks include *Pendulum-v1*, *Hopper-v5*, *HalfCheetah-v5*, *Walker2d-v5*, and *Ant-v5*. Together, these environments provide a progression of increasing difficulty in terms of system dimensionality, control coordination, and stability requirements. This diversity enables evaluation of the proposed framework across a spectrum of control challenges ranging from simple stabilization to high-dimensional locomotion.

Each environment provides a continuous state space representing the physical configuration of the agent and a continuous action space corresponding to actuator torques applied to the system. Table 2 summarizes the dimensionality of the state and action spaces for each environment along with brief qualitative descriptions of their complexity.

Table 2 Continuous control benchmark environments used for evaluation

Environment	State Dim	Action Dim	Comments
Pendulum-v1	3	1	Low complexity stabilization task
Hopper-v5	11	3	Unstable single-leg locomotion
HalfCheetah-v5	17	6	Planar multi-joint running task
Walker2d-v5	17	6	Bipedal locomotion with balance constraints
Ant-v5	105	8	High-dimensional quadruped locomotion

The environments span a broad range of control difficulties. *Pendulum-v1* represents a low-dimensional stabilization task with a single control input. *Hopper-v5* introduces underactuated dynamics and balance constraints that make learning more unstable. *HalfCheetah-v5* and *Walker2d-v5* involve coordinated multi-joint locomotion requiring sustained forward motion while maintaining stability. Finally, *Ant-v5* represents a significantly higher-dimensional problem involving coordinated control of multiple limbs and complex contact dynamics.

The increasing dimensionality and control coordination requirements across these environments provide a useful basis for evaluating algorithm scalability and learning efficiency. In later analysis, these environments are also used to construct a relative complexity axis for comparing the performance of baseline reinforcement learning algorithms with the proposed GANGSTER framework.

5.2 Baselines

To evaluate the effectiveness of the proposed GANGSTER framework, its performance was compared against two widely used off-policy deep reinforcement learning algorithms: Twin Delayed Deep Deterministic Policy Gradient (TD3) [2] and Soft Actor-Critic (SAC) [3]. These algorithms represent strong baseline methods for continuous control and are commonly used as benchmarks in MuJoCo-based reinforcement learning tasks.

In this study, *vanilla* implementations of TD3 and SAC were used without any additional exploration mechanisms, curriculum strategies, or architectural modifications. Both algorithms were trained using the same environments, reward structures, and training budgets as the proposed GANGSTER framework.

Using standard implementations of TD3 and SAC allows the evaluation to isolate the impact of the proposed subgoal generation and prioritization framework while ensuring that comparisons are made against well-established reinforcement learning baselines.

5.3 Metrics

To evaluate the performance of the proposed GANGSTER framework, several metrics were computed from the episodic extrinsic reward trajectory observed during training. These metrics are designed to capture not only the final performance of the agent but also the speed of learning, stability of the learned policy, and consistency of task completion.

Let R_i denote the extrinsic reward obtained at episode i, where $i \in \{1, 2, \ldots, N\}$ and N is the total number of training episodes. A sliding evaluation window of size w is used to determine when the task is considered solved. The mean reward within the window beginning at episode i is defined as

$$\bar{R}_i = \frac{1}{w} \sum_{k=i}^{i+w-1} R_k \tag{20}$$

where w denotes the number of consecutive episodes used for evaluation.

5.3.1 Time to Solve (TTS)

The Time to Solve (TTS) measures how quickly an agent reaches a satisfactory performance level. Let τ denote a predefined performance threshold. The environment is considered solved when the sliding window average reward exceeds this threshold. The Time to Solve is therefore defined as

$$\text{TTS} = \min\{i + w \mid \bar{R}_i > \tau\} \tag{21}$$

where i represents the first episode index for which the windowed average reward exceeds the threshold τ. If no such episode exists, the task is considered unsolved.

5.3.2 Post-Solution Stability

After the solution point is reached, policy stability is evaluated using the standard deviation of rewards obtained after episode TTS. Let the set of post-solution rewards be

$$\mathcal{R}_{post} = \{R_{\text{TTS}}, R_{\text{TTS}+1}, \ldots, R_N\}. \tag{22}$$

The post-solution stability metric is defined as

$$\sigma_{post} = \sqrt{\frac{1}{|\mathcal{R}_{post}|} \sum_{R_i \in \mathcal{R}_{post}} (R_i - \mu_{post})^2} \tag{23}$$

where μ_{post} is the mean reward over the post-solution episodes and $|\mathcal{R}_{post}|$ denotes the number of episodes in the post-solution set.

5.3.3 Post-Solution Average Reward

The long-term performance of the learned policy is measured by the average reward after the solution point. This metric is defined as

$$\mu_{post} = \frac{1}{|\mathcal{R}_{post}|} \sum_{R_i \in \mathcal{R}_{post}} R_i \tag{24}$$

where $\mathcal{R}_{post}$ represents the set of rewards obtained after the solution episode.

5.3.4 Consistency Score

The consistency score measures how reliably the agent maintains performance above the solution threshold after the task is solved. It is defined as the percentage of post-solution episodes for which the reward remains above the threshold τ:

$$C = \frac{1}{|\mathcal{R}_{post}|} \sum_{R_i \in \mathcal{R}_{post}} \mathbb{I}(R_i > \tau) \times 100 \tag{25}$$

where $\mathbb{I}(\cdot)$ denotes the indicator function, which returns 1 if the condition is true and 0 otherwise.

5.3.5 Final 50-Episode Average

To summarize the final policy performance independent of the early learning phase, the average reward over the final N_f episodes is computed as

$$R_{final} = \frac{1}{N_f} \sum_{i=N-N_f+1}^{N} R_i \tag{26}$$

where N_f denotes the number of final evaluation episodes. In this work, $N_f = 50$.

Together, these metrics provide a comprehensive evaluation of reinforcement learning performance. Time to Solve measures learning efficiency, post-solution stability and consistency characterize policy robustness, and the Final 50-Episode Average captures the ultimate performance level achieved by the trained agent. These metrics enable systematic comparison between baseline reinforcement learning algorithms and the proposed GANGSTER framework.

5.4 Evaluation Protocol

All experiments were conducted using a fixed training horizon of $E = 500$ episodes per run. This episode budget was selected to provide sufficient time for exploration and policy convergence while maintaining consistent training conditions across all algorithms and environments.

To account for the inherent stochasticity of deep reinforcement learning, each experiment was repeated across $S = 5$ independent Monte Carlo seeds. Each seed corresponds to a different initialization of random number generators controlling environment resets, neural network initialization, replay buffer sampling, and exploration noise. Let $R_e^{(s)}$ denote the episodic reward obtained at episode e under seed s, where

$$e \in \{1, 2, \ldots, E\}, \quad s \in \{1, 2, \ldots, S\}. \tag{27}$$

5.4.1 Monte Carlo Mean Performance

The expected reward trajectory across seeds is estimated by averaging rewards at each episode index. The mean reward at episode e is defined as

$$\bar{R}_e = \frac{1}{S} \sum_{s=1}^{S} R_e^{(s)}. \tag{28}$$

This aggregated trajectory provides an estimate of the expected learning behavior of the algorithm while reducing sensitivity to individual stochastic training runs.

5.4.2 Variance Across Seeds

Variability across training runs is quantified using the sample variance of rewards across seeds:

$$\sigma_e^2 = \frac{1}{S-1} \sum_{s=1}^{S} \left(R_e^{(s)} - \bar{R}_e \right)^2. \tag{29}$$

The corresponding standard deviation σ_e provides a measure of the variability in training outcomes across different random seeds.

5.4.3 Confidence Intervals

To visualize uncertainty in the reward trajectories, confidence intervals are computed using the standard error of the mean:

$$SE_e = \frac{\sigma_e}{\sqrt{S}}. \tag{30}$$

Assuming approximate normality across runs, the $(1 - \alpha)$ confidence interval for the reward at episode e is

$$CI_e = \bar{R}_e \pm z_{\alpha/2} SE_e, \tag{31}$$

where $z_{\alpha/2}$ is the critical value of the standard normal distribution. In this work, 95% confidence intervals are reported.

5.4.4 Reward Curve Smoothing

To improve the interpretability of reward trajectories, learning curves are smoothed using a moving average filter. The smoothed reward $\tilde{R}_e$ at episode e is computed as

$$\tilde{R}_e = \frac{1}{W} \sum_{k=e-W+1}^{e} \bar{R}_k, \tag{32}$$

where W denotes the smoothing window size. This operation reduces high-frequency fluctuations caused by exploration noise while preserving the underlying learning trend.

Together, the use of multiple Monte Carlo seeds, confidence intervals, and smoothed reward trajectories provides a statistically grounded evaluation of algorithm performance. This protocol ensures that reported learning curves reflect consistent trends across runs rather than isolated stochastic outcomes.

6 Results

This section presents the empirical evaluation of the proposed GANGSTER framework. The experiments are designed to examine three key questions: (1) how the different components of the cGAN subgoal loss influence performance, (2) whether the resulting configuration generalizes across environments when combined with standard reinforcement learning algorithms, and (3) how algorithm performance changes as environment complexity increases. To address these questions, we first perform an ablation study on the cGAN loss components to determine effective weighting strategies for diversity, desirability, and reachability. The best-performing configuration identified in this study is then evaluated in a cross-environment comparison against baseline reinforcement learning agents. Finally, we analyze performance trends as a function of environment complexity to examine how the proposed framework scales across increasingly difficult control tasks. The following subsections present the results of these experiments.

6.1 Ablation Study

To understand the contribution of the individual components of the GANGSTER subgoal generation objective, an ablation study was performed on the three loss terms used to guide the conditional GAN: diversity, desirability, and reachability. These terms serve complementary roles in shaping the generated subgoal distribution. The diversity term encourages exploration by promoting coverage of different regions of the state space. The desirability term biases the generator toward states associated with higher expected returns. The reachability term encourages the generation of goals that are feasible for the agent to achieve from its current state.

To evaluate the relative importance of these terms, several weighting configurations were tested. Each configuration assigns weights to the diversity (w_D), desirability (w_V), and reachability (w_R) terms within the generator loss. The configurations evaluated include individual loss terms, pairwise combinations, and an equal-weight configuration in which all three terms contribute equally. For reference, a baseline configuration corresponding to previously used GANGSTER settings and a control case with no GANGSTER mechanism were also included.

The evaluated configurations are summarized in Table 3. For pairwise combinations, the corresponding loss weights were normalized such that their sum remained comparable across configurations.

The ablation experiments were conducted using representative algorithm–environment pairings to evaluate both moderate and high-complexity locomotion tasks. The resulting performance metrics were compared across configurations, and the learning curves were analyzed using the evaluation framework described in Section 5. Figure 3 and Figure 4 illustrate the performance of the primary configurations evaluated in the study.

The results reveal several consistent trends. First, the desirability term plays the most significant role in driving performance improvements, indicating that biasing subgoal generation toward high-value states is critical for effective guidance of the

Table 3 Loss-weight configurations evaluated in the cGAN ablation study

Configuration	Diversity	Desirability	Reachability
Baseline GANGSTER	0.25	2.0	0.5
Diversity Only	1	0	0
Desirability Only	0	1	0
Reachability Only	0	0	1
Diversity + Desirability	0.5	0.5	0
Diversity + Reachability	0.5	0	0.5
Desirability + Reachability	0	0.5	0.5
Equal Weights	0.33	0.33	0.33
No GANGSTER	0	0	0

learning process. Second, the diversity term contributes to improved long-term exploration by encouraging broader coverage of the state space, which can support more robust policy discovery over extended training horizons. Finally, the reachability term primarily contributes to stability by discouraging the generation of subgoals that are infeasible or excessively distant from the agent's current state.

Taken together, these observations suggest that desirability provides the strongest performance signal, while diversity and reachability serve supporting roles that improve exploration breadth and training stability. These findings motivate the weighting strategy adopted for the final GANGSTER configuration used in the subsequent experiments.

6.1.1 Ant TD3 Ablation

The first ablation experiment evaluates the impact of the three GANGSTER subgoal loss components—diversity, desirability, and reachability—when applied to TD3 on the *Ant-v5* environment. Ant represents a relatively high-dimensional locomotion task with complex coordination requirements, making it a useful testbed for evaluating the effect of different subgoal generation strategies.

Figure 3 illustrates the learning curves for the evaluated configurations, while Table 4 summarizes the performance metrics across runs. Several consistent trends emerge from the results.

First, configurations emphasizing the *desirability* term exhibit the strongest performance overall. The desirability-only configuration achieves the highest final reward trajectory among the tested variants and maintains strong performance across multiple evaluation metrics. In particular, desirability-focused configurations achieve competitive time-to-solve values while also maintaining high post-solution reward levels and consistency scores. This suggests that biasing the generator toward states associated with higher value estimates provides a strong guiding signal for the learning process.

Second, the *diversity* term appears to contribute primarily to long-term exploration rather than immediate performance gains. While the diversity-only configuration does not produce the highest final rewards, its learning curve demonstrates steady improvement over the course of training. When combined with desirability, the resulting configurations exhibit improved learning stability and sustained performance growth,

suggesting that diversity helps prevent premature convergence to narrow regions of the state space.

Third, the *reachability* term plays an important stabilizing role. Reachability-focused configurations tend to produce more consistent learning trajectories and improved consistency scores after solving the task. By discouraging the generation of subgoals that are too distant or infeasible, the reachability term helps ensure that generated goals remain actionable for the policy.

Comparing the GANGSTER configurations with the baseline TD3 agent further highlights the benefit of guided subgoal generation. The baseline agent exhibits slower learning progress and lower final reward values relative to several GANGSTER variants. In contrast, configurations incorporating desirability and reachability demonstrate faster improvement and higher final reward levels.

Overall, the ablation study indicates that the desirability term provides the strongest performance signal for subgoal generation, while diversity and reachability contribute complementary benefits related to exploration breadth and training stability. These observations motivate the weighting strategy adopted for the final GANGSTER configuration used in subsequent experiments.

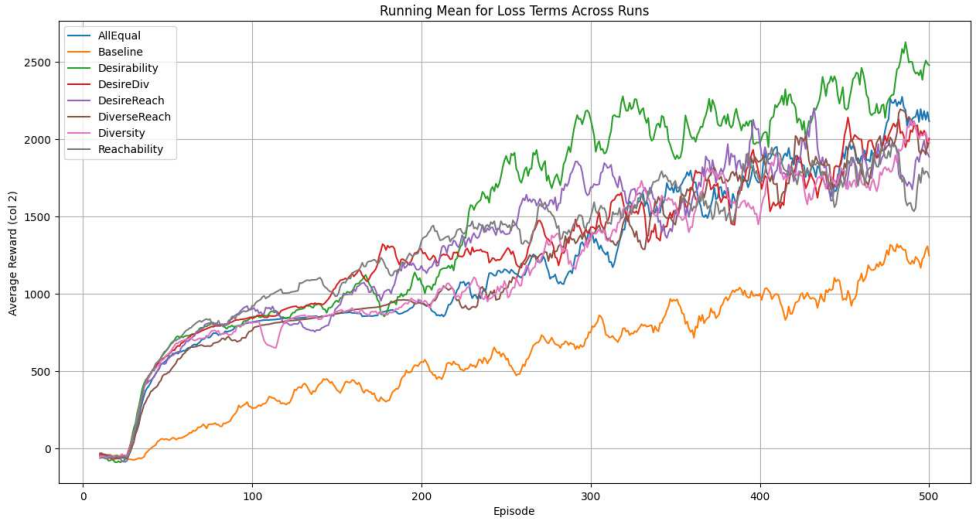


Fig. 3 TD3 Ant Ablation Results

Table 4 TD3 and Ant Ablation Study Results (Mean ± Std Dev)

AllEqual			**DesireReach**		
Metric	Mean	Std Dev	Metric	Mean	Std Dev
Time to Solve (TTS)	258.2	91.7701	Time to Solve (TTS)	149	45.6333
Post-Solution Stability	747.0641	180.5579	Post-Solution Stability	791.3406	160.8014
Post-Solution Average	1654.3001	245.0115	Post-Solution Average	1603.9568	231.5824
Consistency Score (%)	78.5859	2.7275	Consistency Score (%)	72.9311	10.0404
Final 50-Episode Avg	2053.0878	452.5376	Final 50-Episode Avg	1853.1744	318.1925
Baseline			**DiverseReach**		
Time to Solve (TTS)	303.4	58.7081	Time to Solve (TTS)	209.5	50.7174
Post-Solution Stability	**546.5378**	107.5163	Post-Solution Stability	801.3389	65.5918
Post-Solution Average	964.973	217.7545	Post-Solution Average	1748.0701	159.9916
Consistency Score (%)	40.269	13.4662	Consistency Score (%)	77.9085	4.7453
Final 50-Episode Avg	1205.9922	399.2495	Final 50-Episode Avg	1971.0642	542.3324
Desirability			**Diversity**		
Time to Solve (TTS)	**142.8**	34.301	Time to Solve (TTS)	197.6	47.0472
Post-Solution Stability	900.5073	39.1335	Post-Solution Stability	702.8821	76.2118
Post-Solution Average	**1871.5705**	174.5695	Post-Solution Average	1486.9148	340.0892
Consistency Score (%)	77.6287	4.561	Consistency Score (%)	70.691	16.6666
Final 50-Episode Avg	**2393.221**	122.7777	Final 50-Episode Avg	1874.5963	497.381
DesireDiv			**Reachability**		
Time to Solve (TTS)	153	65.4889	Time to Solve (TTS)	136	77.3789
Post-Solution Stability	790.8938	149.7652	Post-Solution Stability	731.5706	75.8196
Post-Solution Average	1564.4356	137.4769	Post-Solution Average	1728.4797	268.5944
Consistency Score (%)	72.3867	7.8378	Consistency Score (%)	**83.973**	5.9218
Final 50-Episode Avg	2002.8054	375.7753	Final 50-Episode Avg	1781.3646	531.8971

6.1.2 HalfCheetah-v5 (SAC) Ablation

The second ablation experiment evaluates the effect of the cGAN loss components when integrated with the Soft Actor-Critic (SAC) algorithm on the *HalfCheetah-v5* environment. Compared to Ant, HalfCheetah represents a moderately complex locomotion task without strict balance constraints, allowing the evaluation to focus more directly on exploration and reward-driven subgoal generation.

Figure 4 illustrates the learning curves for the evaluated configurations, while Table 5 summarizes the corresponding performance metrics. Overall, all GANGSTER configurations outperform the vanilla SAC baseline, indicating that adversarially generated subgoals consistently improve learning efficiency and final policy performance in this environment.

Among the evaluated configurations, the *desirability-focused* variants again demonstrate the strongest overall performance. Configurations emphasizing the desirability term achieve the highest post-solution reward levels and final 50-episode averages, while also maintaining high consistency scores above 97%. These results reinforce the

observation from the TD3–Ant ablation that directing the generator toward high-value states provides the most effective learning signal for subgoal generation.

The *diversity* term again appears to support exploration rather than directly driving peak performance. Diversity-only configurations produce competitive learning curves but generally converge to slightly lower final reward values compared to desirability-dominant configurations. However, when diversity is combined with desirability, the resulting configurations demonstrate smooth learning trajectories and strong final performance, suggesting that diversity helps maintain broader state-space exploration during training.

The *reachability* term continues to contribute primarily to training stability. Configurations incorporating reachability maintain high consistency scores and stable reward trajectories after solving the task, indicating that constraining subgoal feasibility prevents the generator from proposing goals that are excessively distant from the agent's current state distribution.

A notable observation in the SAC experiments is that performance differences between GANGSTER configurations are smaller than those observed in the TD3–Ant experiments. Most configurations converge to relatively high reward levels once learning stabilizes, suggesting that SAC's entropy-driven exploration already provides strong baseline exploration capabilities. Nevertheless, desirability-weighted configurations consistently produce the highest final performance, supporting the conclusion that desirability remains the dominant guiding signal for effective subgoal generation.

Taken together, the SAC–HalfCheetah ablation results confirm the trends observed in the TD3–Ant study. The desirability term provides the primary performance signal, while diversity and reachability contribute complementary benefits related to exploration coverage and training stability. These findings justify the desirability-dominant weighting used in the final GANGSTER configuration evaluated in subsequent experiments.

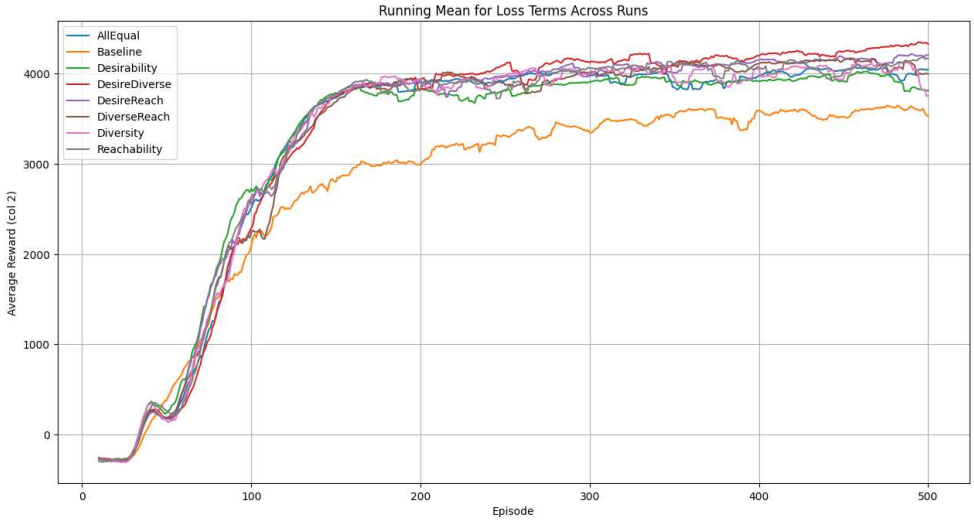


Fig. 4 SAC-HalfCheetah-v5 Ablation Results

Table 5 SAC and HalfCheetah-v5 Ablation Study Results (Mean ± Std Dev)

AllEqual			**DesireReach**		
Metric	Mean	Std Dev	Metric	Mean	Std Dev
Time to Solve (TTS)	60	7.4027	Time to Solve (TTS)	55.4	4.9234
Post-Solution Stability	628.7794	56.9997	Post-Solution Stability	682.0528	122.1186
Post-Solution Average	3745.7641	110.7962	Post-Solution Average	3795.5183	201.5156
Consistency Score (%)	98.7513	0.6086	Consistency Score (%)	98.7679	0.8146
Final 50-Episode Avg	4027.5676	154.5619	Final 50-Episode Avg	4164.9899	283.4855
Baseline			**Reachability**		
Time to Solve (TTS)	**54.4**	9.4361	Time to Solve (TTS)	59.4	3.7202
Post-Solution Stability	684.3634	113.4991	Post-Solution Stability	652.5213	87.3159
Post-Solution Average	3158.6944	71.6953	Post-Solution Average	3784.9193	168.7964
Consistency Score (%)	97.4438	1.4739	Consistency Score (%)	98.8904	0.5095
Final 50-Episode Avg	3599.856	86.5419	Final 50-Episode Avg	4118.1041	206.189
Desirability			**Diversity**		
Time to Solve (TTS)	54.6	5.5353	Time to Solve (TTS)	58	6
Post-Solution Stability	**605.9483**	51.541	Post-Solution Stability	734.7313	102.6857
Post-Solution Average	3672.9494	82.2035	Post-Solution Average	3754.4183	154.6005
Consistency Score (%)	98.8102	0.5818	Consistency Score (%)	97.9593	0.6791
Final 50-Episode Avg	3913.4137	93.2526	Final 50-Episode Avg	4016.3078	303.7451
DesireDiverse					
Time to Solve (TTS)	65.4	8.5697			
Post-Solution Stability	642.0721	86.8493			
Post-Solution Average	**3891.3403**	123.1284			
Consistency Score (%)	**99.1971**	0.6169			
Final 50-Episode Avg	**4298.1669**	136.4557			

6.2 Comparison of Ablation Results

The two ablation studies provide complementary perspectives on the role of the cGAN loss components across different reinforcement learning algorithms and environments. While the TD3–Ant and SAC–HalfCheetah experiments differ in both control difficulty and underlying learning dynamics, several consistent trends emerge from the results.

First, the desirability term consistently provides the strongest performance signal for subgoal generation. In both experiments, configurations incorporating desirability produce the highest reward trajectories and strongest final performance metrics. This observation suggests that guiding the generator toward states associated with higher expected returns is the most effective mechanism for improving learning performance.

Second, the diversity term contributes primarily to exploration rather than directly driving peak performance. Diversity-only configurations generally achieve competitive learning curves but rarely outperform desirability-focused configurations. However, when combined with desirability, the resulting variants exhibit smoother learning trajectories and improved long-term performance, indicating that diversity supports broader state-space exploration during training.

Third, the reachability term plays an important stabilizing role. Configurations incorporating reachability tend to produce more consistent learning behavior and higher post-solution stability metrics. By discouraging subgoals that are excessively distant or infeasible from the agent's current state, the reachability constraint helps maintain stable training dynamics.

Although the SAC–HalfCheetah experiments show smaller performance differences between configurations than the TD3–Ant experiments, the overall trends remain consistent across both studies. Desirability-dominant configurations consistently achieve the strongest performance, while diversity and reachability provide complementary benefits related to exploration coverage and training stability.

Based on these observations, the final GANGSTER configuration used in the subsequent experiments emphasizes the desirability term while retaining smaller contributions from diversity and reachability. This weighting reflects the dominant role of desirability in guiding subgoal generation while preserving the stabilizing and exploratory benefits of the remaining components.

6.3 Cross-Environment Results

To evaluate whether the benefits of the proposed GANGSTER framework generalize beyond the environments used in the ablation studies, a cross-environment evaluation was conducted across a set of standard continuous-control benchmarks. In these experiments, vanilla SAC and TD3 agents were compared against their GANGSTER-augmented counterparts using the same training budgets and experimental conditions. The environments evaluated include *Pendulum-v1*, *Hopper-v5*, *HalfCheetah-v5*, *Walker2d-v5*, and *Ant-v5*, which span a range of control difficulties and system dimensionalities. For the GANGSTER-enhanced agents, the loss-weighting configuration identified in the ablation studies was used for all experiments. Each algorithm–environment pairing was trained for 500 episodes and repeated across five

independent Monte Carlo seeds to account for stochastic variability in training. Performance was evaluated using the metrics defined in Section 5.3, allowing comparison of learning efficiency, stability, and final policy quality across environments.

6.3.1 TD3 Cross Environment Results

TD3 results are listed below:

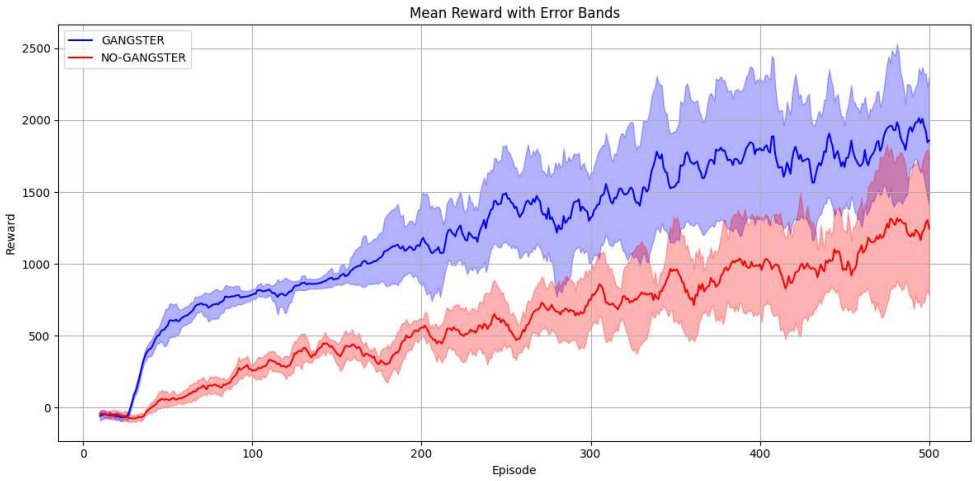


Fig. 5 TD3 Ant Cross Environment Results

Table 6 TD3 vs TD3 + GANGSTER on Ant-v5 (Cross-Environment Evaluation). Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	**209**	303.4	-1.72	0.136	-1.09
Post-Solution Stability	650.08	**546.54**	1.25	0.249	0.79
Post-Solution Average	**1581.28**	964.97	4.51	0.002	2.85
Consistency Score (%)	**80.37**	40.27	6.60	0.002	4.18
Final 50-Episode Avg	**1850.62**	1205.99	2.72	0.027	1.72

Table 6 summarizes the cross-environment comparison between vanilla TD3 and TD3 augmented with the GANGSTER framework on the Ant-v5 environment. The results indicate that the GANGSTER-enhanced agent achieves substantially stronger performance across most evaluation metrics.

The GANGSTER agent reduces the time-to-solve relative to the baseline, although this difference does not reach statistical significance ($p = 0.136$). More substantial improvements appear in the reward-based metrics. In particular, the GANGSTER

agent achieves significantly higher post-solution average reward and final performance levels, with large effect sizes ($d = 2.85$ and $d = 1.72$, respectively). The consistency score also increases dramatically, indicating that the learned policy maintains high reward levels far more reliably after solving the task.

The only metric where the baseline performs better is post-solution stability, indicating slightly lower reward variance after convergence. However, this stability occurs at substantially lower reward levels and therefore does not translate into stronger overall performance.

Overall, these results suggest that the GANGSTER framework significantly improves the final performance and reliability of TD3 on the Ant environment, while also accelerating the learning process relative to the baseline agent.

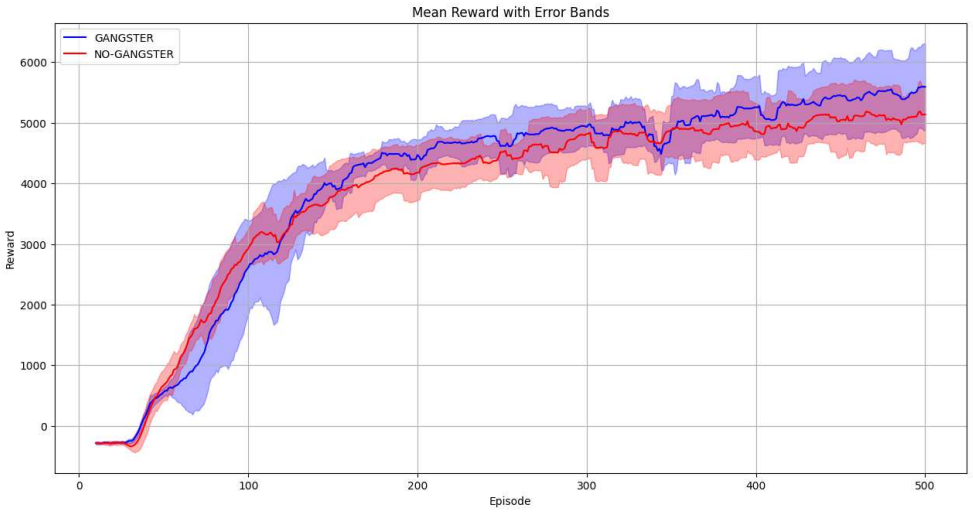


Fig. 6 TD3 HalfCheetah-v5 Cross Environment Results

Table 7 TD3 vs TD3 + GANGSTER on HalfCheetah-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	59	**45.8**	2.81	0.028	1.78
Post-Solution Stability	1096.84	**1033.26**	0.44	0.672	0.28
Post-Solution Average	**4572.14**	4298.52	1.22	0.258	0.77
Consistency Score (%)	97.62	**98.43**	-0.88	0.417	-0.56
Final 50-Episode Avg	**5486.28**	5101.92	1.07	0.321	0.67

Table 7 summarizes the comparison between vanilla TD3 and TD3 augmented with the GANGSTER framework on the HalfCheetah-v5 environment. In contrast to the Ant results, both agents achieve strong performance on this task, reflecting the relative stability of the HalfCheetah-v5 environment for off-policy actor–critic methods.

The baseline TD3 agent solves the task slightly faster than the GANGSTER variant, reaching the solution threshold earlier on average. This difference is statistically significant ($p = 0.028$). The baseline also exhibits slightly lower post-solution variance and marginally higher consistency scores, indicating stable policy behavior once the task has been solved.

However, the GANGSTER-enhanced agent achieves higher overall reward levels after convergence. In particular, the post-solution average reward and final 50-episode average are higher for the GANGSTER configuration, suggesting improved long-horizon policy quality. Although these improvements do not reach statistical significance in this environment, the effect sizes indicate moderate performance gains.

Overall, these results suggest that while HalfCheetah can already be solved reliably by vanilla TD3, the GANGSTER framework provides modest improvements in final reward performance, albeit at the cost of slightly slower convergence.

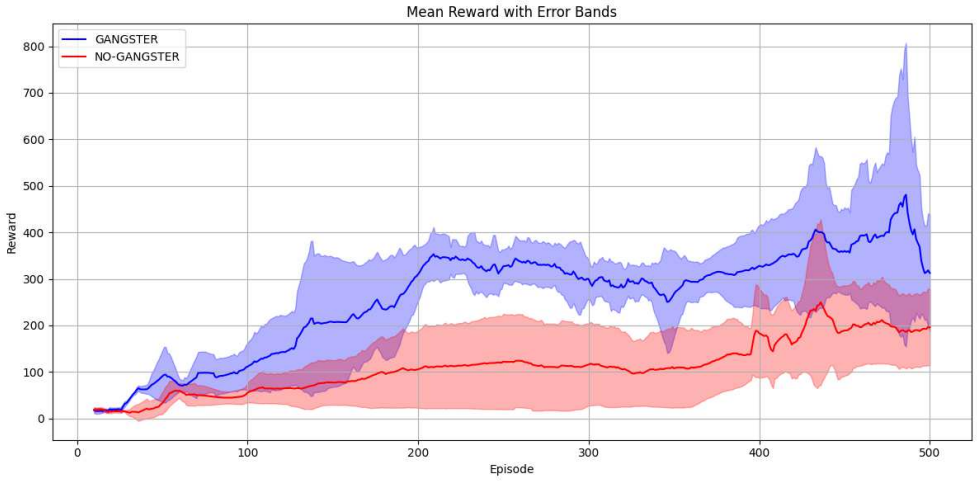


Fig. 7 TD3 Hopper Cross Environment Results

Table 8 presents the comparison between vanilla TD3 and TD3 augmented with the GANGSTER framework on the Hopper-v5 environment. In contrast to the HalfCheetah results, the Hopper environment shows clear and consistent improvements when the GANGSTER framework is applied.

The most pronounced improvement occurs in learning efficiency. The GANGSTER agent solves the task substantially faster, reducing the average time-to-solve from 400

Table 8 TD3 vs TD3 + GANGSTER on Hopper-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	**159.4**	400.5	-16.12	1.0×10^{-5}	-10.20
Post-Solution Stability	**94.43**	136.49	-1.20	0.271	-0.76
Post-Solution Average	**333.89**	258.31	2.78	0.050	1.76
Consistency Score (%)	**64.81**	28.78	2.82	0.033	1.78
Final 50-Episode Avg	**386.70**	196.24	2.33	0.059	1.48

episodes to approximately 159 episodes. This difference is highly statistically significant ($p < 10^{-4}$) and corresponds to an extremely large effect size ($d = -10.20$), indicating a dramatic acceleration of the learning process.

Beyond faster convergence, the GANGSTER-enhanced agent also achieves higher reward levels after solving the task. Both post-solution average reward and the final 50-episode average are substantially higher than the baseline TD3 agent. The GANGSTER configuration also maintains significantly higher consistency scores, indicating that the learned policy remains above the solution threshold more reliably across episodes.

Post-solution stability is also improved, with the GANGSTER agent exhibiting lower reward variance after convergence, although this difference does not reach statistical significance.

Overall, these results indicate that the GANGSTER framework substantially improves both the learning speed and final policy performance of TD3 on the Hopper environment.

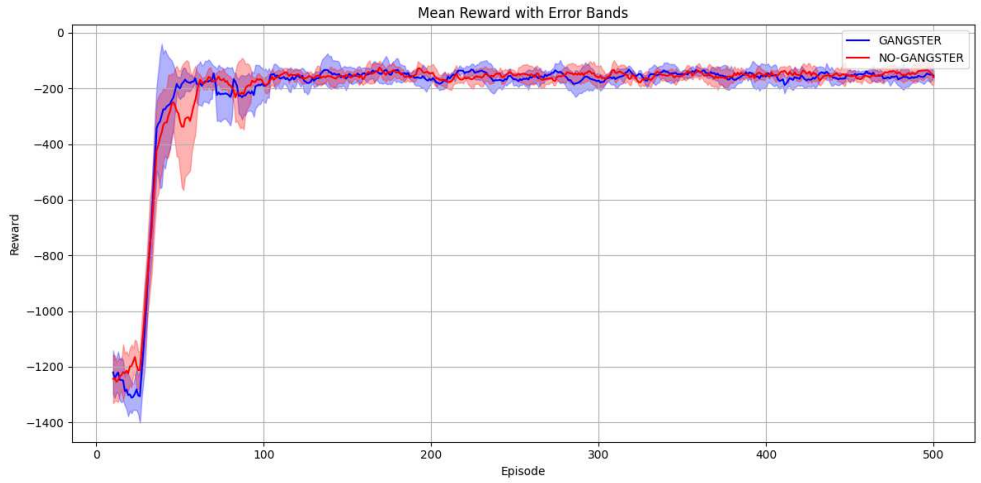


Fig. 8 TD3 Pendulum Cross Environment Results

Table 9 TD3 vs TD3 + GANGSTER on Pendulum-v1. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	**27.2**	34.2	-1.28	0.253	-0.81
Post-Solution Stability	113.22	**101.13**	1.09	0.323	0.69
Post-Solution Average	-160.84	**-156.83**	-1.48	0.183	-0.94
Consistency Score (%)	65.25	**67.73**	-1.98	0.084	-1.25
Final 50-Episode Avg	-155.95	**-150.38**	-0.85	0.431	-0.54

Table 9 presents the comparison between vanilla TD3 and TD3 augmented with the GANGSTER framework on the Pendulum-v1 environment. Similar to the SAC results, this low-complexity task is solved effectively by the baseline algorithm without requiring additional exploration mechanisms.

The GANGSTER-enhanced agent achieves a slightly lower time-to-solve; however, this improvement is not statistically significant. In contrast, the baseline TD3 agent demonstrates marginally better post-solution reward, consistency, and final performance, although none of these differences reach statistical significance.

Post-solution stability is also slightly better for the baseline agent, indicating lower variance after convergence. Overall, the results suggest that the additional exploration introduced by the GANGSTER framework does not provide measurable benefits in this simple environment and may slightly degrade performance.

These findings are consistent with an exploration-driven mechanism that is most beneficial in more complex or unstable environments, where naive exploration strategies are insufficient.

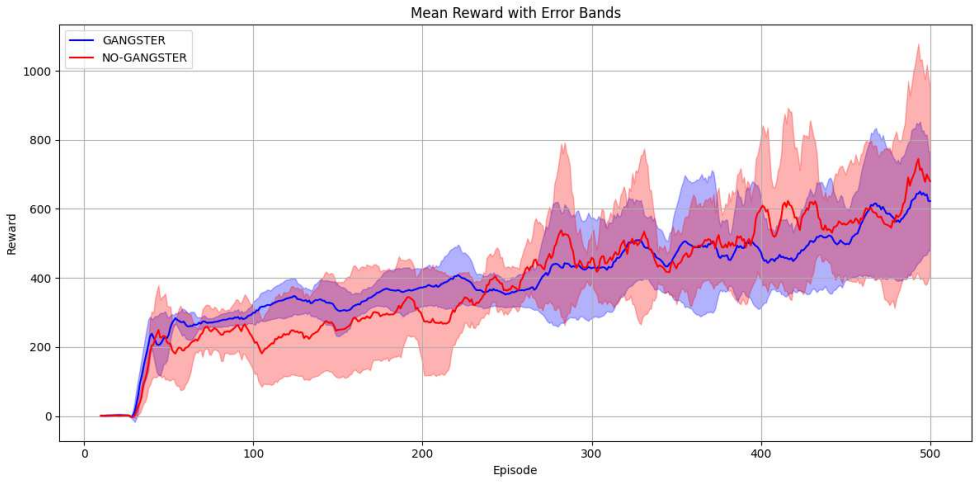


Fig. 9 TD3 Walker2D Cross Environment Results

Table 10 TD3 vs TD3 + GANGSTER on Walker2d-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	**29.8**	75.0	-1.82	0.141	-1.15
Post-Solution Stability	**128.46**	220.97	-1.86	0.103	-1.17
Post-Solution Average	415.72	**435.00**	-0.39	0.706	-0.25
Consistency Score (%)	**78.95**	76.49	0.29	0.777	0.19
Final 50-Episode Avg	597.26	**620.53**	-0.18	0.862	-0.11

Table 10 presents the comparison between vanilla TD3 and TD3 augmented with the GANGSTER framework on the Walker2d-v5 environment. The results indicate a mixed performance profile, with the GANGSTER-enhanced agent improving convergence speed and stability, while the baseline achieves slightly higher final reward.

Runs that did not satisfy the predefined solution criterion within the 500-episode training budget were classified as unsolved. Their TTS values were right-censored at 500 episodes for descriptive summaries and visualization. The GANGSTER agent converges faster, achieving a lower time-to-solve, and demonstrates improved post-solution stability, indicating reduced variance after convergence. These trends suggest that the GANGSTER framework promotes more stable learning dynamics in this environment.

However, the baseline TD3 agent achieves slightly higher post-solution average reward and final performance. These differences are small and not statistically significant, indicating broadly comparable performance between the two methods.

Overall, these results suggest that in intermediate-complexity environments such as Walker2d, the GANGSTER framework improves learning stability and convergence speed, while maintaining competitive final performance relative to the baseline.

6.3.2 SAC Cross Environment Results

SAC Results are listed below:

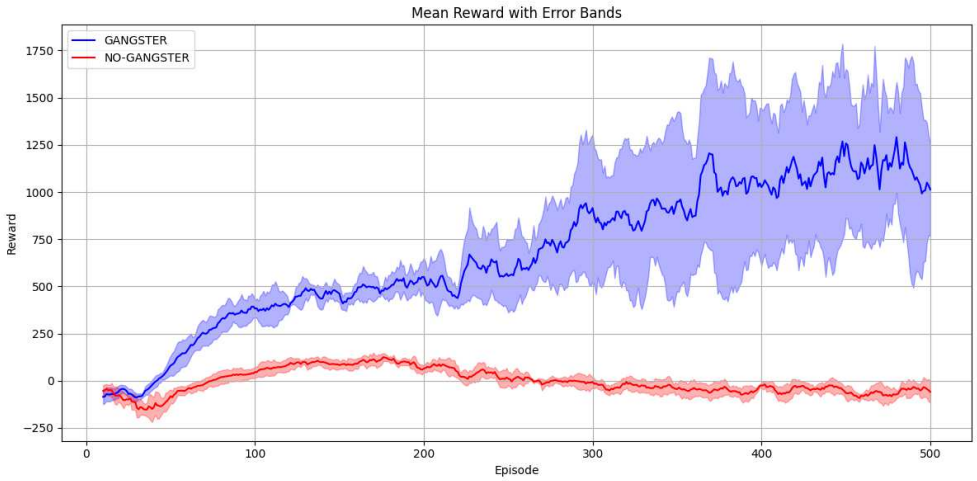


Fig. 10 SAC Ant Cross Environment Results

Table 11 SAC vs SAC + GANGSTER on Ant-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	**288.4**	500	-6.54	0.0028	-4.14
Post-Solution Stability	604.33	**7.39**	9.33	0.0007	5.90
Post-Solution Average	**1010.20**	114.84	6.53	0.0029	4.13
Consistency Score (%)	**39.41**	0.00	4.22	0.0134	2.67
Final 50-Episode Avg	**1110.05**	-61.24	7.55	0.0016	4.77

Table 11 presents the comparison between vanilla SAC and SAC augmented with the GANGSTER framework on the Ant-v5 environment. The results show substantial performance improvements across most metrics when the GANGSTER framework is incorporated.

Runs that did not satisfy the predefined solution criterion within the 500-episode training budget were classified as unsolved. Their TTS values were right-censored at 500 episodes for descriptive summaries and visualization.

The GANGSTER-enhanced SAC agent significantly reduces the time-to-solve, converging much faster than the baseline ($p < 0.01$, $d = -4.14$). More notably, the reward-based metrics exhibit dramatic improvements. The baseline SAC agent struggles to achieve meaningful performance on this task, as reflected by low post-solution rewards and a zero consistency score. In contrast, the GANGSTER-enhanced agent achieves substantially higher post-solution rewards, improved consistency, and significantly higher final performance, all with large effect sizes.

The only metric where the baseline outperforms GANGSTER is post-solution stability, where the baseline exhibits lower variance. However, this stability occurs at very low reward levels and does not correspond to meaningful task performance.

Overall, these results indicate that the GANGSTER framework enables SAC to effectively learn and maintain performance on the Ant environment, where the baseline SAC agent fails to consistently solve the task.

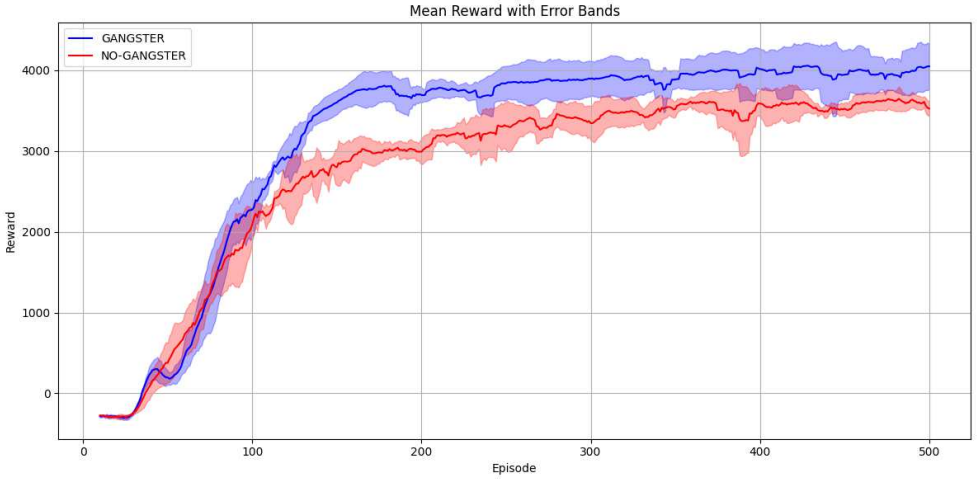


Fig. 11 SAC HalfCheetah Cross Environment Results

Table 12 SAC vs SAC + GANGSTER on HalfCheetah-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	60.6	**54.4**	1.23	0.259	0.78
Post-Solution Stability	**624.41**	684.36	-1.05	0.334	-0.66
Post-Solution Average	**3674.05**	3158.69	5.88	0.0018	3.72
Consistency Score (%)	**99.17**	97.44	2.56	0.058	1.62
Final 50-Episode Avg	**3988.08**	3599.86	3.28	0.022	2.07

Table 12 presents the comparison between vanilla SAC and SAC augmented with the GANGSTER framework on the HalfCheetah-v5 environment. Both agents achieve strong performance on this task; however, the GANGSTER-enhanced agent demonstrates consistent improvements in post-convergence performance.

The baseline SAC agent converges slightly faster, achieving a lower time-to-solve, although this difference is not statistically significant. In contrast, the GANGSTER-enhanced agent exhibits improved post-solution behavior, achieving higher reward levels and lower variance after convergence.

In particular, the GANGSTER agent significantly improves post-solution average reward and final performance ($p < 0.01$), with large effect sizes ($d = 3.72$ and $d = 2.07$,

respectively). The consistency score is also higher, indicating that the policy maintains high reward levels more reliably across episodes.

Overall, these results suggest that while SAC already performs well on HalfCheetah, the GANGSTER framework improves the quality and reliability of the learned policy without significantly affecting convergence speed.

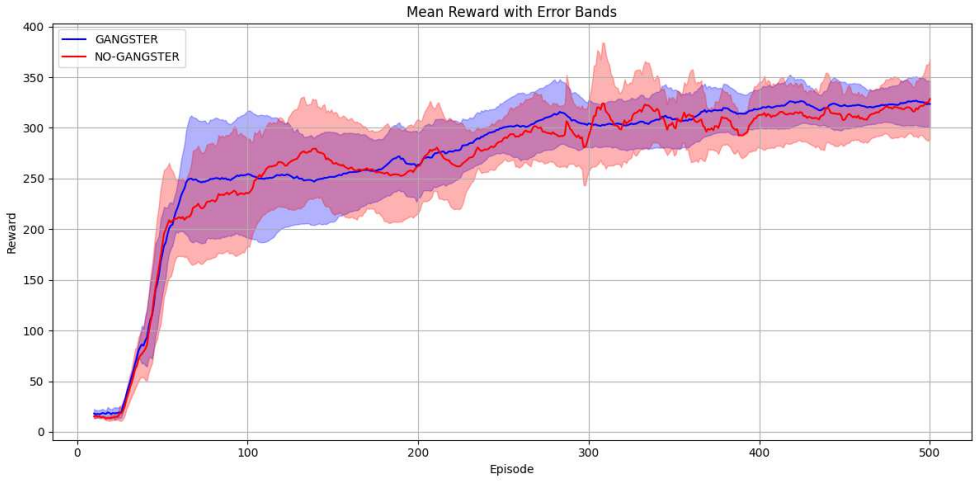


Fig. 12 SAC Hopper Cross Environment Results

Table 13 SAC vs SAC + GANGSTER on Hopper-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	163	**127**	0.51	0.621	0.33
Post-Solution Stability	**20.98**	47.98	-2.63	0.048	-1.66
Post-Solution Average	**318.88**	307.25	1.67	0.145	1.05
Consistency Score (%)	**82.05**	62.89	3.10	0.024	1.96
Final 50-Episode Avg	**323.21**	318.09	0.34	0.744	0.21

Table 13 presents the comparison between vanilla SAC and SAC augmented with the GANGSTER framework on the Hopper-v5 environment. The results indicate that while the baseline SAC agent converges slightly faster, the GANGSTER-enhanced agent demonstrates improved post-solution performance and stability.

The baseline achieves a lower time-to-solve, although this difference is not statistically significant. In contrast, the GANGSTER agent exhibits significantly lower post-solution variance ($p < 0.05$), indicating more stable policy behavior after convergence.

In addition, the GANGSTER-enhanced agent achieves higher consistency scores, maintaining performance above the solution threshold more reliably across episodes. While improvements in post-solution average reward and final performance are modest and not statistically significant, the effect sizes suggest consistent gains in policy quality.

Overall, these results suggest that for moderately challenging environments such as Hopper, the GANGSTER framework improves stability and reliability of learned policies, while introducing a small trade-off in convergence speed.



Fig. 13 SAC Pendulum Cross Environment Results

Table 14 SAC vs SAC + GANGSTER on Pendulum-v1. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	86.6	**46.6**	2.49	0.054	1.57
Post-Solution Stability	**97.81**	103.97	-0.57	0.595	-0.36
Post-Solution Average	-157.78	**-153.28**	-1.20	0.269	-0.76
Consistency Score (%)	66.05	**68.47**	-1.94	0.096	-1.23
Final 50-Episode Avg	-152.12	**-137.50**	-2.18	0.065	-1.38

Table 14 presents the comparison between vanilla SAC and SAC augmented with the GANGSTER framework on the Pendulum-v1 environment. In this low-complexity setting, the baseline SAC agent outperforms the GANGSTER-enhanced variant across most metrics.

The baseline agent converges substantially faster, achieving a lower time-to-solve, with the difference approaching statistical significance. In addition, the baseline demonstrates slightly higher post-solution reward, consistency, and final performance, although these differences do not reach statistical significance.

The GANGSTER-enhanced agent achieves marginally lower post-solution variance; however, this improvement is small and does not translate into higher overall performance.

Overall, these results indicate that in very low-complexity environments, the additional exploration introduced by the GANGSTER framework can hinder convergence and does not provide measurable benefits. This behavior is consistent with an exploration mechanism that is most beneficial in more complex or unstable environments, where naive exploration strategies are insufficient.

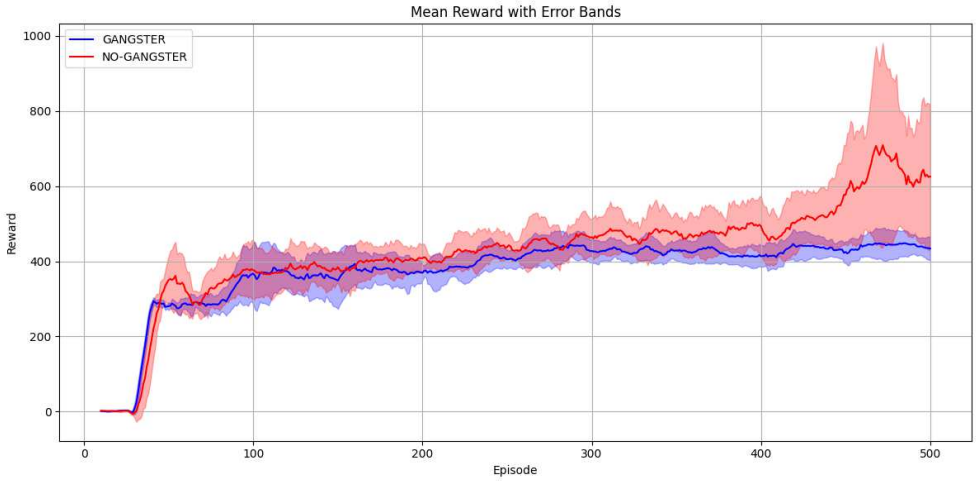


Fig. 14 SAC Walker2D Cross Environment Results

Table 15 SAC vs SAC + GANGSTER on Walker2d-v5. Lower values are better for TTS and Post-Solution Stability, while higher values are better for the remaining metrics

Metric	GANGSTER	Baseline	t-stat	p-value	Cohen's d
Time to Solve (TTS)	**29.4**	32.8	-2.19	0.068	-1.39
Post-Solution Stability	**73.10**	139.68	-2.72	0.046	-1.72
Post-Solution Average	397.01	**454.63**	-2.19	0.074	-1.39
Consistency Score (%)	89.24	**92.58**	-1.56	0.169	-0.99
Final 50-Episode Avg	442.28	**641.22**	-2.54	0.060	-1.61

Table 15 presents the comparison between vanilla SAC and SAC augmented with the GANGSTER framework on the Walker2d-v5 environment. The results indicate a

mixed performance profile, with the GANGSTER-enhanced agent improving learning efficiency and stability, while the baseline SAC agent achieves higher final reward levels.

The GANGSTER-enhanced agent converges faster, achieving a lower time-to-solve, and demonstrates significantly improved post-solution stability ($p < 0.05$), indicating reduced variance after convergence. These results suggest that the exploration strategy introduced by GANGSTER facilitates more stable learning dynamics in this environment.

However, the baseline SAC agent achieves higher post-solution average reward, consistency, and final performance. Although some of these differences approach statistical significance, they do not consistently reach conventional thresholds.

Overall, these results suggest that for intermediate-complexity environments such as Walker2d, the GANGSTER framework improves learning stability and convergence speed, but may trade off some degree of final performance. This behavior is consistent with an exploration mechanism that prioritizes stable policy discovery over aggressive reward maximization.

6.3.3 Cross-Environment TD3 and SAC Analysis

We compare the impact of the GANGSTER framework across TD3 and SAC to assess whether its effects are algorithm-dependent or consistent across off-policy actor–critic methods. Overall, the results indicate that GANGSTER provides qualitatively similar benefits for both algorithms, while interacting differently with their underlying exploration dynamics.

For both TD3 and SAC, the most significant improvements occur in more complex or unstable environments such as Hopper and Ant. In these settings, GANGSTER consistently improves post-solution reward, consistency, and, in several cases, convergence speed. These gains are particularly pronounced in the Ant environment, where both TD3 and SAC show substantial improvements in final performance and reliability, suggesting that GANGSTER effectively enhances exploration in high-dimensional and sparse-reward settings.

In moderately complex environments such as Walker2d, the impact of GANGSTER is more nuanced. For both TD3 and SAC, GANGSTER tends to improve stability and, in some cases, convergence speed, while producing mixed results in final reward. This suggests that the framework biases learning toward more stable policy discovery, which may not always correspond to maximal reward optimization.

In simpler environments such as HalfCheetah and Pendulum, both algorithms already perform strongly without additional exploration mechanisms. In these cases, GANGSTER provides only modest improvements or may slightly degrade performance, typically by slowing convergence. This behavior is consistent across both TD3 and SAC, indicating that the additional exploration introduced by GANGSTER is unnecessary in low-complexity settings.

A key difference between the algorithms is that SAC, due to its entropy-regularized objective, already incorporates a degree of stochastic exploration. As a result, the relative gains from GANGSTER are generally smaller for SAC in simpler environments compared to TD3. However, in more complex tasks, both algorithms benefit

substantially, indicating that GANGSTER complements rather than replaces inherent exploration mechanisms.

Overall, these results suggest that the effectiveness of GANGSTER is primarily driven by environment complexity rather than the choice of underlying algorithm. The framework provides the greatest benefit in environments where exploration is a limiting factor, while introducing minor trade-offs in simpler settings where exploration is less critical.

6.4 Computational Overhead Analysis

In addition to evaluating learning performance, we measure the computational overhead introduced by the GANGSTER framework. Since GANGSTER augments the base reinforcement learning algorithm with conditional subgoal generation, candidate ranking, and intrinsic reward computation, it is expected to increase the average wall-clock time required per environment step. To quantify this cost, we compare the average time per training step for agents trained with and without GANGSTER across the benchmark environments.

Table 16 reports the average time per step for each environment. The percent delta is computed as

$$\Delta_{\%} = \frac{T_{\text{GANGSTER}} - T_{\text{Baseline}}}{T_{\text{Baseline}}} \times 100, \tag{33}$$

where T_{GANGSTER} denotes the average time per step using the GANGSTER framework and T_{Baseline} denotes the corresponding average time per step without GANGSTER.

Table 16 Average compute time per training step with and without GANGSTER

Environment	GANGSTER	No GANGSTER	Percent Delta
Pendulum-v1	0.0896	0.0406	121%
Walker2d-v5	0.1137	0.0590	93%
Hopper-v5	0.0881	0.0279	216%
Ant-v5	0.0865	0.0450	92%
HalfCheetah-v5	0.0560	0.0387	45%

The results show that GANGSTER increases the average computation time per training step across all environments. The largest relative overhead occurs in Hopper-v5, where the average step time increases by approximately 216%, while the smallest relative overhead occurs in HalfCheetah-v5, where the increase is approximately 45%. This confirms that the proposed framework introduces a nontrivial computational cost relative to the baseline agents.

This overhead should be interpreted as a trade-off rather than as an isolated measure of inefficiency. GANGSTER introduces additional computation in exchange for structured exploration and improved learning behavior in several environments, particularly those where exploration and coordination are more difficult. The added cost

reflects the generation and evaluation of candidate subgoals rather than changes to the underlying actor–critic update rules. As a result, the overhead is modular and could potentially be reduced through less frequent subgoal generation, smaller candidate sets, or optimized generator inference.

These results also motivate the following complexity-based analysis. Since GANGSTER introduces additional per-step computation, its usefulness depends on whether the performance gains become more pronounced as environment complexity increases. The next section therefore examines whether the benefits of structured subgoal generation scale with task complexity.

6.5 Performance vs Complexity

In addition to environment-specific comparisons, we analyze how algorithm performance changes as task complexity increases. The goal of this analysis is to determine whether the proposed GANGSTER framework exhibits different scaling behavior compared to baseline reinforcement learning algorithms when applied to progressively more complex environments.

To enable this analysis, each benchmark environment is assigned a scalar complexity score defined as

$$C(\text{env}) = \log(d_s + d_a) + \lambda B(\text{env}), \tag{34}$$

where d_s and d_a denote the dimensionality of the state and action spaces, respectively. The logarithmic dimensionality term captures the increase in representational and control complexity as environments grow larger. The term $B(\text{env})$ represents a balance difficulty factor that reflects qualitative control challenges such as instability and coordination requirements that are not fully captured by dimensionality alone. The parameter λ controls the relative influence of this balance factor. This composite score serves as a heuristic proxy for relative task difficulty across the benchmark environments.

For each evaluation metric, the mean performance value obtained in each environment is paired with the corresponding complexity score. This produces a set of (C_i, M_i) pairs, where C_i denotes the complexity of environment i and M_i denotes the observed performance metric.

6.5.1 Trend Estimation Using Ordinary Least Squares

To quantify how performance changes as complexity increases, a linear trend is estimated using ordinary least squares (OLS) regression. For each metric, a linear model of the form

$$M_i = \beta_0 + \beta_1 C_i + \epsilon_i \tag{35}$$

is fitted, where M_i is the performance metric for environment i, C_i is the corresponding complexity score, β_0 is the intercept, β_1 is the slope coefficient, and ϵ_i represents the residual error.

The slope coefficient β_1 provides a measure of how the metric changes as environment complexity increases. Positive slopes indicate that the metric tends to increase

with complexity, while negative slopes indicate degradation as tasks become more difficult.

Separate regressions are computed for baseline algorithms and their GANGSTER-enhanced counterparts. Comparing the resulting slope coefficients provides insight into how the proposed framework affects scaling behavior across environments.

6.5.2 Interpreting Scaling Behavior

The interpretation of the slope coefficient depends on the directionality of the metric being analyzed. For reward-based metrics such as final average reward, larger values correspond to better performance. In this case, a larger or less negative slope indicates that performance degrades more slowly as complexity increases, suggesting improved scalability.

For efficiency metrics such as Time to Solve (TTS), smaller values correspond to better performance. In this case, a smaller slope indicates that the number of episodes required to solve the task grows more slowly with increasing complexity.

By comparing slope estimates between baseline algorithms and GANGSTER-enhanced agents, we evaluate whether the proposed framework alters the relationship between task complexity and learning performance. This analysis provides an additional perspective on algorithm behavior that complements the environment-specific results presented in the following section.

6.5.3 TD3 Complexity Results

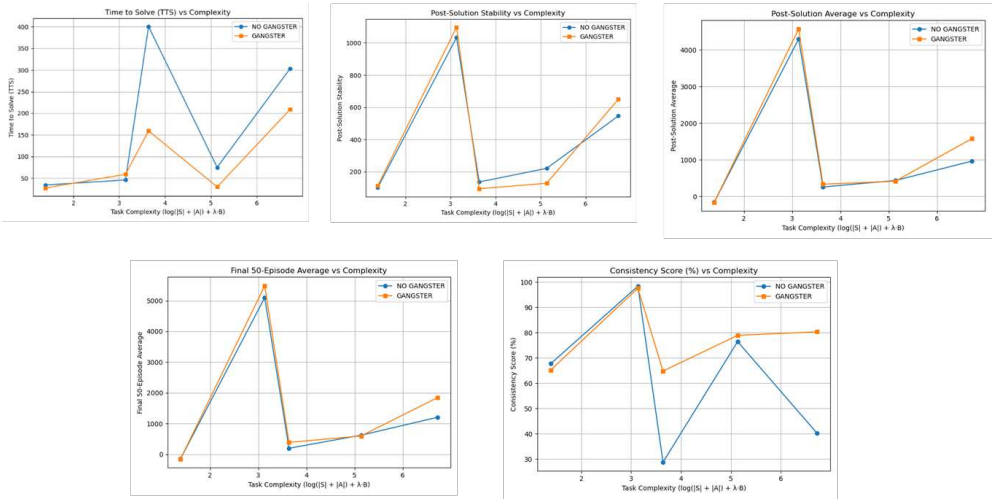


Fig. 15 Performance metrics as a function of task complexity for TD3 with and without the GANGSTER framework. Lower values are better for Time-to-Solve (TTS) and Post-Solution Stability, while higher values are better for the remaining metrics

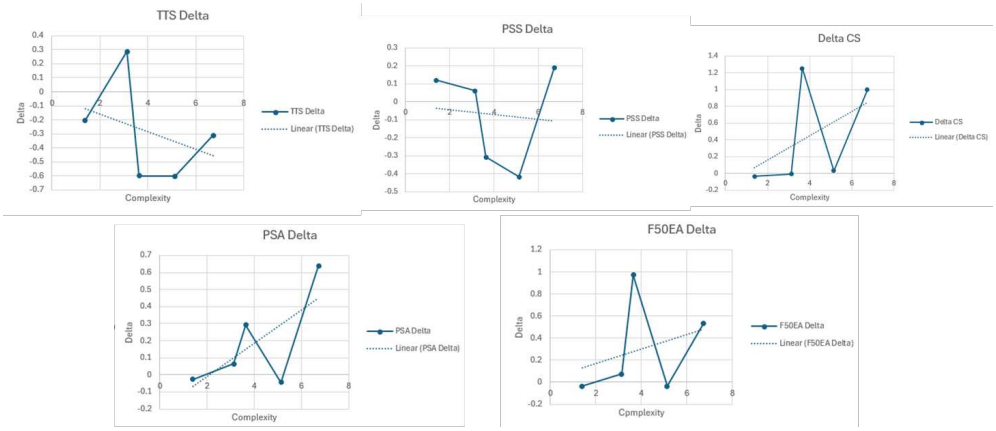


Fig. 16 Performance deltas (Δ) between GANGSTER and baseline TD3 as a function of environment complexity. Linear trend lines are shown for each metric

Table 17 OLS regression results for performance deltas (Δ) as a function of environment complexity. Positive slopes indicate increasing benefit of GANGSTER with complexity (for reward-based metrics), while negative slopes indicate improvements for metrics where lower is better (e.g., TTS, PSS)

Metric	Slope (β_1)	Intercept (β_0)	R^2
Time to Solve (TTS)	-0.0622	-0.0372	0.119
Post-Solution Stability (PSS)	-0.0129	-0.0197	0.009
Post-Solution Average (PSA)	0.0969	-0.2032	0.469
Consistency Score (CS)	0.1456	-0.1360	0.223
Final 50-Episode Average (F50EA)	0.0663	0.0357	0.305

To quantify how the impact of the GANGSTER framework scales with environment complexity, we analyze the performance deltas between GANGSTER and baseline agents across environments. For each metric, we fit a linear model relating the observed performance difference to the complexity measure.

As shown in Table 17 and Figure 16, the results reveal a consistent trend: the benefits of GANGSTER increase with task complexity, particularly for reward-based metrics. Positive slopes are observed for post-solution average reward (PSA), consistency score (CS), and final performance (F50EA), indicating that GANGSTER yields larger performance gains in more complex environments.

In contrast, time-to-solve (TTS) and post-solution stability (PSS) exhibit negative slopes. Since lower values are preferred for these metrics, this indicates that GANGSTER reduces convergence time and improves stability as complexity increases. However, these effects are weaker and exhibit lower coefficients of determination, suggesting greater variability across environments.

The strength of the regression fits varies by metric. Reward-based metrics show the strongest relationships with complexity (e.g., $R^2 \approx 0.47$ for PSA), while convergence-related metrics exhibit weaker correlations. This suggests that the primary advantage of GANGSTER lies in improving policy quality and reliability rather than uniformly accelerating convergence.

Importantly, the delta plots indicate that gains are minimal or even negative in low-complexity environments, while becoming strongly positive in higher-complexity settings. This supports the hypothesis that GANGSTER primarily enhances exploration, providing the greatest benefit when standard exploration strategies are insufficient.

Overall, these results demonstrate that the effectiveness of the GANGSTER framework scales with environment complexity, with the most pronounced improvements occurring in higher-dimensional and more challenging control tasks.

6.5.4 SAC Complexity Results

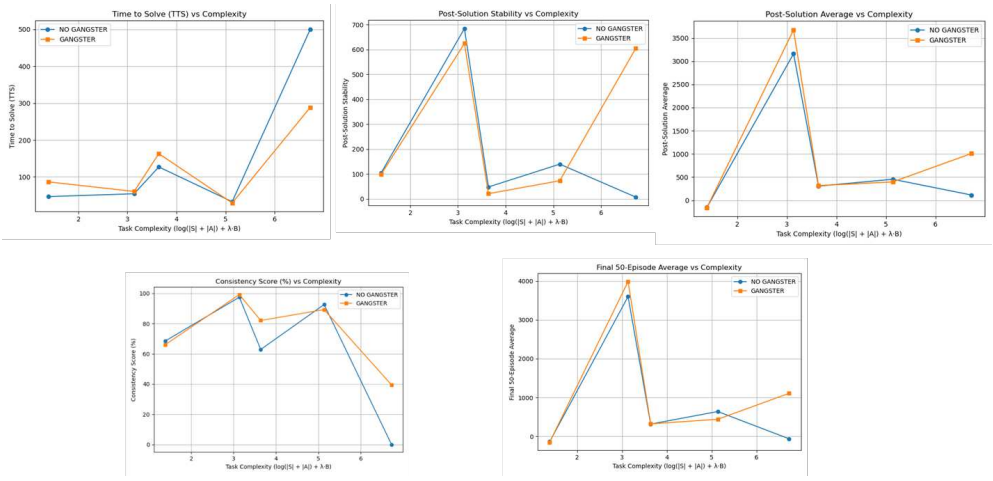


Fig. 17 Complexity analysis for SAC comparing GANGSTER and baseline across environments.

To evaluate how the impact of GANGSTER scales with task complexity under SAC, we analyze performance deltas between GANGSTER and baseline agents across environments. Linear models are fit to each metric as a function of environment complexity.

As shown in Table 18 and Figure 18, the results indicate a strong positive relationship between complexity and the benefits of GANGSTER for most reward-based metrics. In particular, post-solution average reward (PSA), consistency score (CS), and final performance (F50EA) all exhibit positive slopes with moderate explanatory power ($R^2 \approx 0.55$), suggesting that GANGSTER provides increasingly meaningful gains as task complexity grows.

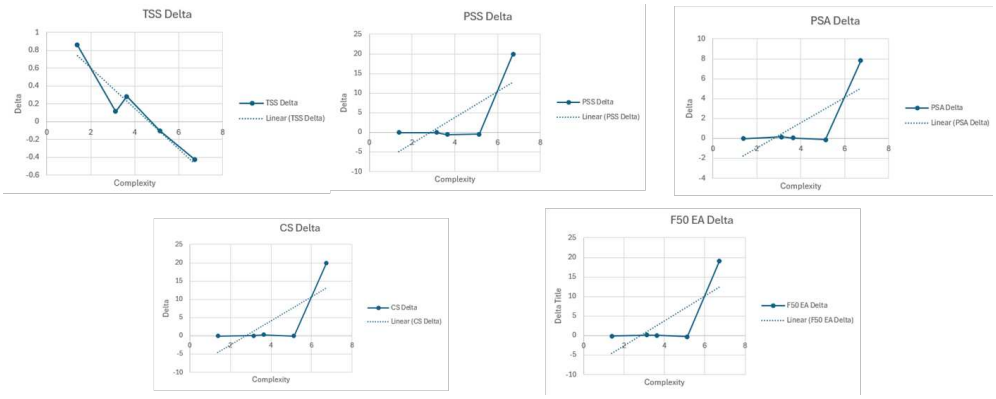


Fig. 18 Performance deltas (Δ) between GANGSTER and baseline SAC as a function of environment complexity. Linear trend lines are shown for each metric: Time to Solve (TTS), Post-Solution Stability (PSS), Post-Solution Average (PSA), Consistency Score (CS), and Final 50-Episode Average (F50EA).

Table 18 OLS regression results for SAC performance deltas (Δ) as a function of environment complexity. Positive slopes indicate increasing benefit of GANGSTER with complexity (for reward-based metrics), while negative slopes indicate improvements for metrics where lower is better (e.g., TTS)

Metric	Slope (β_1)	Intercept (β_0)	R^2
Time to Solve (TTS)	-0.2263	1.0521	0.9213
Post-Solution Stability (PSS)	3.3066	-9.4795	0.5450
Post-Solution Average (PSA)	1.2779	-3.5494	0.5531
Consistency Score (CS)	3.3082	-9.1985	0.5656
Final 50-Episode Average (F50EA)	3.1574	-8.8781	0.5556

Time to solve (TTS) exhibits a strong negative slope ($\beta_1 = -0.2263$) with a high coefficient of determination ($R^2 = 0.92$), indicating a consistent reduction in convergence time as complexity increases. This suggests that GANGSTER improves exploration efficiency in more challenging environments, allowing the agent to reach successful policies more quickly.

Post-solution stability (PSS) shows a positive slope, indicating that stability may degrade slightly as complexity increases under GANGSTER. However, this effect is moderate and less pronounced than the gains observed in reward-based metrics, suggesting a trade-off between aggressive exploration and stability.

Overall, the results demonstrate that GANGSTER scales favorably with environment complexity under SAC, with the most consistent improvements observed in convergence speed and policy performance. The high R^2 for TTS further supports the claim that GANGSTER provides structured exploration benefits that become increasingly important in more complex environments.

6.5.5 Complexity Analysis Discussion

To better understand how the proposed GANGSTER framework scales across algorithms, we compare the observed complexity trends for both TD3 and SAC. While both methods benefit from GANGSTER, the nature and consistency of these improvements differ across algorithms.

For TD3, the impact of GANGSTER is evident but exhibits higher variability across environments. Reward-based metrics, including post-solution average (PSA), consistency score (CS), and final performance (F50EA), generally show positive trends with increasing complexity, indicating that GANGSTER provides greater benefit in more challenging tasks. However, the associated regression fits are moderate, suggesting that these improvements are less consistent and more environment-dependent. Similarly, convergence-related metrics such as time to solve (TTS) and post-solution stability (PSS) show weaker and noisier trends, reflecting the sensitivity of TD3 to exploration dynamics.

In contrast, SAC demonstrates more structured and consistent scaling behavior under GANGSTER. The most notable result is observed in time to solve (TTS), which exhibits a strong negative slope with a high coefficient of determination, indicating that GANGSTER consistently reduces convergence time as task complexity increases. Reward-based metrics also show clear positive trends with moderate explanatory power, suggesting that GANGSTER improves both learning efficiency and final policy quality in a more predictable manner compared to TD3. While post-solution stability (PSS) shows a slight degradation with increasing complexity, this effect is relatively small and is outweighed by gains in performance and consistency.

Comparing the two algorithms reveals an important distinction: GANGSTER provides more stable and predictable benefits when paired with SAC, while its effects on TD3, although positive, are more variable. This difference is likely due to SAC's entropy-regularized objective, which inherently promotes exploration and may better complement the structured subgoal generation and prioritization mechanisms introduced by GANGSTER. In contrast, TD3 relies more heavily on deterministic policy updates, making it more sensitive to the additional exploratory signals.

Overall, these results suggest that GANGSTER enhances exploration in a way that scales with environment complexity for both algorithms, but does so more reliably under SAC. The framework's benefits are most pronounced in higher-complexity environments, supporting the hypothesis that structured subgoal-driven exploration becomes increasingly valuable as task difficulty grows.

7 Discussion

The experimental results demonstrate that GANGSTER provides consistent improvements in learning efficiency, stability, and final performance across a range of continuous-control environments. However, these benefits are not uniform across tasks and are most pronounced under specific conditions related to environment complexity and reward structure.

7.1 When GANGSTER Helps Most

GANGSTER shows the strongest performance gains in environments characterized by high dimensionality, complex dynamics, and delayed reward signals. In tasks such as Ant-v5 and Walker2d-v5, where coordinated multi-joint control and long-horizon credit assignment are required, structured subgoal generation provides meaningful guidance for exploration. In these settings, standard exploration mechanisms such as action noise or entropy regularization are insufficient to efficiently discover high-reward behaviors.

The observed improvements in Time to Solve and final performance suggest that GANGSTER effectively reduces the burden of undirected exploration by introducing intermediate objectives that guide the agent toward promising regions of the state space. Furthermore, the complexity analysis indicates that these benefits scale with increasing task difficulty, supporting the hypothesis that structured subgoal discovery becomes more valuable as environment complexity grows.

7.2 When GANGSTER Helps Less

In lower-complexity environments such as Pendulum-v1, the benefits of GANGSTER are limited. These tasks are typically solved quickly by baseline algorithms due to dense reward signals and relatively simple dynamics. In such cases, the introduction of subgoal generation provides little additional benefit and may introduce unnecessary overhead.

This behavior highlights an important characteristic of the framework: GANGSTER is not intended to outperform baseline methods in trivial settings, but rather to address regimes where exploration and credit assignment become bottlenecks. As a result, performance gains should be interpreted in the context of task complexity rather than absolute improvement across all environments.

7.3 Role of Structured Subgoal Components

The ablation results provide insight into the functional roles of the three structured loss components: desirability, reachability, and diversity.

Desirability plays the most significant role in driving performance improvements. By biasing subgoal generation toward states associated with high expected return, this component directly aligns subgoal selection with the long-term objective of the agent. Across environments, configurations that emphasize desirability consistently achieve faster convergence and higher final rewards.

Reachability contributes primarily to training stability. By penalizing subgoals that are distant or infeasible, it prevents the generator from proposing unrealistic targets that could destabilize learning. This effect is particularly important in high-dimensional environments, where unconstrained subgoal generation may lead to erratic or unproductive exploration.

Diversity supports broader state-space coverage and mitigates collapse to a narrow set of subgoals. While its impact on early learning speed is less pronounced, it contributes to improved long-term performance and consistency by encouraging

exploration of multiple trajectories. This is especially beneficial in environments with multiple viable strategies or complex dynamics.

Together, these components form a complementary system in which desirability drives progress, reachability enforces feasibility, and diversity maintains exploration breadth.

7.4 Trade-offs and Limitations

The introduction of GANGSTER involves several trade-offs. First, the framework increases computational cost due to the additional training of the generator and discriminator networks. This overhead is nontrivial, particularly in environments where baseline methods already achieve satisfactory performance.

Second, the framework introduces additional hyperparameters, including the weighting coefficients for the structured loss components and the intrinsic reward scaling factor. While the ablation study provides guidance on effective configurations, sensitivity to these parameters may affect reproducibility and require tuning for new environments.

Third, GANGSTER may be less effective in tasks with dense rewards or simple dynamics, where the marginal benefit of structured exploration is limited. In such cases, the added complexity of the framework may not justify its use.

Finally, the current evaluation is limited to simulated MuJoCo environments. While these benchmarks provide a standard testbed for continuous-control reinforcement learning, further validation is required to assess performance in real-world or safety-critical settings.

7.5 Summary

Overall, the results suggest that GANGSTER is most beneficial in settings where exploration is a primary bottleneck and task complexity is high. By introducing structured, interpretable subgoal generation without modifying the underlying reinforcement learning algorithm, the framework provides a scalable and modular approach to improving exploration in continuous-control domains.

8 Limitations and Future Work

While the proposed GANGSTER framework demonstrates consistent improvements in learning efficiency and performance, several limitations should be acknowledged.

First, the framework introduces additional computational overhead due to the training of the generator and discriminator networks. Compared to standard off-policy reinforcement learning methods, this results in increased training time and resource requirements. Although this overhead is manageable in simulated environments, it may become a limiting factor in large-scale or real-time applications.

Second, the performance of GANGSTER depends on the choice of hyperparameters, particularly the weighting coefficients associated with desirability, reachability, and diversity, as well as the intrinsic reward scaling factor. While the ablation study

provides insight into effective configurations, sensitivity to these parameters may affect reproducibility and require environment-specific tuning.

Third, the current evaluation is limited to MuJoCo-based continuous-control benchmarks. While these environments provide a standardized and widely accepted testbed, they do not fully capture the variability, noise, and constraints present in real-world systems. As such, further validation is necessary to assess the robustness and generalization of the proposed approach beyond simulation.

In addition, the framework assumes that meaningful subgoals can be represented directly within the state space. This assumption may not hold in environments with high-dimensional observations, such as raw visual inputs, where latent representations may be required for effective subgoal generation.

Several directions for future work emerge from these limitations. One promising extension is the development of adaptive or dynamic weighting strategies for the structured loss components, allowing the relative importance of desirability, reachability, and diversity to evolve during training. Such mechanisms could improve robustness across environments with varying characteristics.

Another direction involves integrating the framework with model-based reinforcement learning methods. Combining generative subgoal discovery with learned dynamics models may enable more efficient planning and further improve sample efficiency, particularly in environments with sparse rewards. Another extension of this work woud be the applicabiity of GANGSTER to On-Policy reinforcement learning algorithms. To date, only off-policy algorithms have been integrated with the framework.

Extending the approach to real-world robotic systems represents an important step toward practical deployment. This would require addressing issues such as sensor noise, safety constraints, and limited data availability, which are not present in simulated benchmarks.

Finally, the proposed framework may be extended to multi-agent settings, where coordinated subgoal generation could facilitate cooperation or competition among agents. Exploring how structured subgoal discovery interacts with multi-agent dynamics presents an interesting avenue for future research.

Overall, while GANGSTER introduces additional complexity, it provides a flexible and modular approach to structured exploration, and these limitations highlight opportunities for further refinement and broader application.

9 Conclusion

This work introduced GANGSTER, a generative subgoal framework designed to improve exploration and learning efficiency in continuous-control reinforcement learning. By modeling subgoal discovery as a conditional generative process and incorporating structured objectives for desirability, reachability, and diversity, the proposed approach provides a principled mechanism for guiding exploration without modifying the underlying reinforcement learning algorithm.

The experimental results demonstrate that GANGSTER improves sample efficiency, policy stability, and final performance across multiple MuJoCo benchmark

environments. These improvements are most pronounced in higher-complexity tasks, where standard exploration strategies struggle with long-horizon credit assignment.

In addition, the framework was shown to integrate effectively with both TD3 and SAC, indicating that its benefits are not tied to a specific algorithmic implementation. The analysis of performance as a function of environment complexity further suggests that structured subgoal generation becomes increasingly valuable as task difficulty grows.

Overall, GANGSTER provides a modular and scalable approach to structured exploration in continuous-control domains. By introducing interpretable and complementary mechanisms for subgoal generation, the framework offers a practical pathway for improving reinforcement learning performance in environments where efficient exploration remains a fundamental challenge.

Statements and Declarations

Funding

The authors declare that no funds, grants, or other financial support were received for the preparation of this manuscript.

Competing Interests

The authors have no relevant financial or non-financial interests to disclose.

Author Contributions

William Rudnisky contributed to the conceptualization, methodology, software development, experimental investigation, data analysis, visualization, and preparation of the original manuscript. Yong Wang contributed to the conceptualization, methodology, supervision, and review and editing of the manuscript. Both authors read and approved the final manuscript.

Data Availability

This study did not use any primary or secondary dataset. All experiments were run in simulated reinforcement learning environments (Gymnasium MuJoCo continuous-control tasks), which are publicly available at https://gymnasium.farama.org/environments/mujoco/. The simulated training data and results generated during the current study are available from the corresponding author on reasonable request.

Code Availability

The source code used to implement the algorithms and conduct the experiments is available from the corresponding author upon reasonable request.

Ethics Approval

Not applicable. This article does not contain any studies involving human participants or animals performed by any of the authors.

Consent to Participate

Not applicable.

Consent for Publication

Not applicable.

Clinical Trial Number

Not applicable.

References

[1] Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, 2nd edn. MIT Press, Cambridge, MA (2018)

[2] Fujimoto, S., Hoof, H., Meger, D.: Addressing function approximation error in actor-critic methods. In: Proceedings of the 35th International Conference on Machine Learning, pp. 1587–1596 (2018). https://proceedings.mlr.press/v80/fujimoto18a.html

[3] Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the 35th International Conference on Machine Learning, pp. 1861–1870 (2018). https://proceedings.mlr.press/v80/haarnoja18b.html

[4] Sutton, R.S., Precup, D., Singh, S.: Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. Artificial Intelligence **112**(1–2), 181–211 (1999). https://doi.org/10.1016/S0004-3702(99)00052-1

[5] Barto, A.G., Mahadevan, S.: Recent advances in hierarchical reinforcement learning. Discrete Event Dynamic Systems **13**, 341–379 (2003). https://doi.org/10.1023/A:1025696116075

[6] Nachum, O., Gu, S.S., Lee, H., Levine, S.: HIRO: Data-efficient hierarchical reinforcement learning. In: Advances in Neural Information Processing Systems (2018). https://arxiv.org/abs/1805.08296

[7] Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Abbeel, P., Zaremba, W.: Hindsight experience replay. In: Advances in Neural Information Processing Systems (2017). https://arxiv.org/abs/1707.01495

[8] Pathak, D., Agrawal, P., Efros, A.A., Darrell, T.: Curiosity-driven exploration by self-supervised prediction. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (2017). https://arxiv.org/abs/1705.05363

[9] Burda, Y., Edwards, H., Storkey, A., Klimov, O.: Exploration by random network distillation. In: Proceedings of the 7th International Conference on Learning Representations (2019). https://openreview.net/forum?id=H1lJJnR5Ym

[10] Todorov, E., Erez, T., Tassa, Y.: MuJoCo: A physics engine for model-based control. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5026–5033 (2012). https://doi.org/10.1109/IROS.2012.6386109

[11] Bacon, P.-L., Harb, J., Precup, D.: The option-critic architecture. In: Proceedings of the AAAI Conference on Artificial Intelligence (2017). https://doi.org/10.1609/aaai.v31i1.10916

[12] Vezhnevets, A.S., Osindero, S., Schaul, T., Heess, N., Jaderberg, M., Silver, D., Kavukcuoglu, K.: FeUdal networks for hierarchical reinforcement learning. In: Proceedings of the International Conference on Machine Learning (2017). https://proceedings.mlr.press/v70/vezhnevets17a.html

[13] Levy, A., Konidaris, G., Platt, R., Saenko, K.: Learning multi-level hierarchies with hindsight. In: Proceedings of the International Conference on Learning Representations (2019). https://openreview.net/forum?id=ryzECoAcY7

[14] Chane-Sane, E., Schmid, C., Laptev, I.: Goal-Conditioned reinforcement learning with imagined subgoals. In: Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 1430–1440 (2021). https://proceedings.mlr.press/v139/chane-sane21a.html

[15] Li, S., Zheng, L., Wang, J., Zhang, C.: Learning subgoal representations with slow dynamics. In: International Conference on Learning Representations (2021). https://openreview.net/forum?id=wxRwhSdORKG

[16] Li, S., Zhang, J., Wang, J., Yu, Y., Zhang, C.: Active hierarchical exploration with stable subgoal representation learning. In: International Conference on Learning Representations (2022). https://openreview.net/forum?id=sNuFKTMktcY

[17] Kim, J., Seo, Y., Shin, J.: Landmark-Guided subgoal generation in hierarchical reinforcement learning. In: Advances in Neural Information Processing Systems, vol. 34 (2021). https://openreview.net/forum?id=IWhFd34QSSj

[18] Gürtler, N., Büchler, D., Martius, G.: Hierarchical reinforcement learning with timed subgoals. In: Advances in Neural Information Processing Systems, vol. 34, pp. 21732–21743 (2021). https://openreview.net/forum?id=96ULbah4DC

[19] Lee, S., Kim, J., Jang, I., Kim, H.J.: DHRL: A graph-based approach for long-horizon and sparse hierarchical reinforcement learning. In: Advances in Neural Information Processing Systems, vol. 35 (2022). https://arxiv.org/abs/2210.05150

[20] Zhang, Q., Yang, Y., Ruan, J., Xiong, X., Xing, D., Xu, B.: Balancing exploration and exploitation in hierarchical reinforcement learning via latent landmark graphs. In: 2023 International Joint Conference on Neural Networks (2023). https://doi.org/10.1109/IJCNN54540.2023.10190993

[21] Bougie, N., Ichise, R.: Fast and slow curiosity for high-level exploration in reinforcement learning. Applied Intelligence **51**, 1086–1107 (2021). https://doi.org/10.1007/s10489-020-01849-3

[22] Bougie, N., Ichise, R.: Hierarchical learning from human preferences and curiosity. Applied Intelligence **52**, 7459–7479 (2022). https://doi.org/10.1007/s10489-021-02726-3

[23] Yuan, M., Li, B., Jin, X., Zeng, W.: Automatic intrinsic reward shaping for exploration in deep reinforcement learning. In: Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 40531–40554 (2023). https://proceedings.mlr.press/v202/yuan23c.html

[24] Wang, V.H., Wang, T., Yang, W., Kämäräinen, J.-K., Pajarinen, J.: Probabilistic subgoal representations for hierarchical reinforcement learning. In: Proceedings of the 41st International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 235, pp. 51755–51770 (2024). https://proceedings.mlr.press/v235/wang24bx.html

[25] Li, Y., Wang, Y., Tan, X.: Highly valued subgoal generation for efficient goal-conditioned reinforcement learning. Neural Networks **181**, 106825 (2025). https://doi.org/10.1016/j.neunet.2024.106825

[26] Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning. In: Proceedings of the 4th International Conference on Learning Representations (2016). https://arxiv.org/abs/1509.02971

[27] Houthooft, R., Chen, X., Duan, Y., Schulman, J., De Turck, F., Abbeel, P.: VIME: Variational information maximizing exploration. In: Advances in Neural Information Processing Systems (2016). https://arxiv.org/abs/1605.09674

[28] Bellemare, M.G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., Munos, R.: Unifying count-based exploration and intrinsic motivation. In: Advances in Neural Information Processing Systems (2016). https://arxiv.org/abs/1606.01868

[29] Fortunato, M., Azar, M.G., Piot, B., Menick, J., Osband, I., Graves, A., Mnih,

V., Munos, R., Hassabis, D., Pietquin, O., Blundell, C., Legg, S.: Noisy networks for exploration. In: Proceedings of the International Conference on Learning Representations (2018). https://arxiv.org/abs/1706.10295

[30] Wu, L., Chen, K.: Goal exploration augmentation via pre-trained skills for sparse-reward long-horizon goal-conditioned reinforcement learning. Machine Learning **113**, 2527–2557 (2024). https://doi.org/10.1007/s10994-023-06503-w

[31] Schaul, T., Horgan, D., Gregor, K., Silver, D.: Universal value function approximators. In: Proceedings of the International Conference on Machine Learning, pp. 1312–1320 (2015). https://proceedings.mlr.press/v37/schaul15.html

[32] Hu, E.S., Chang, R., Rybkin, O., Jayaraman, D.: Planning goals for exploration. In: The Eleventh International Conference on Learning Representations (2023). https://openreview.net/forum?id=6qeBuZSo7Pr

[33] Castanet, N., Sigaud, O., Lamprier, S.: Stein variational goal generation for adaptive exploration in Multi-Goal reinforcement learning. In: Proceedings of the 40th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 202, pp. 3714–3731 (2023). https://proceedings.mlr.press/v202/castanet23a.html

[34] Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in Neural Information Processing Systems, pp. 2672–2680 (2014). https://arxiv.org/abs/1406.2661

[35] Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014). https://arxiv.org/abs/1411.1784

[36] Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein GAN. In: Proceedings of the International Conference on Machine Learning (2017). https://arxiv.org/abs/1701.07875

[37] Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: InfoGAN: Interpretable representation learning by information maximizing generative adversarial nets. In: Advances in Neural Information Processing Systems (2016). https://arxiv.org/abs/1606.03657

[38] Haarnoja, T., Tang, H., Abbeel, P., Levine, S.: Reinforcement learning with deep energy-based policies. In: Proceedings of the International Conference on Machine Learning (2017). https://proceedings.mlr.press/v70/haarnoja17a.html

[39] Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., Levine, S.: QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. In: Proceedings

of the 2nd Conference on Robot Learning. Proceedings of Machine Learning Research, vol. 87, pp. 651–673 (2018). https://proceedings.mlr.press/v87/kalashnikov18a.html

[40] Duan, Y., Mao, W., Zhu, H.: Learning world models for unconstrained goal navigation. In: Advances in Neural Information Processing Systems, vol. 37, pp. 59236–59265 (2024). https://arxiv.org/abs/2411.02446

[41] Liu, Y., Liu, X.: Adventurer: Exploration with BiGAN for deep reinforcement learning. Applied Intelligence **55**(10), 726 (2025). https://doi.org/10.1007/s10489-025-06600-4

[42] Hong, W., Zhu, M., Liu, M., Zhang, W., Zhou, M., Yu, Y., Sun, P.: Generative adversarial exploration for reinforcement learning. In: Proceedings of the First International Conference on Distributed Artificial Intelligence, pp. 1–10 (2019). https://doi.org/10.1145/3356464.3357706

[43] Sarmad, M., Lee, H., Kim, Y.M.: RL-GAN-Net: A reinforcement learning agent controlled GAN network. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (2019). https://openaccess.thecvf.com/content_CVPR_2019/html/Sarmad_RL-GAN-Net_A_Reinforcement_Learning_Agent_Controlled_GAN_Network_for_Real-Time_CVPR_2019_paper.html

[44] Yu, L., Zhang, W., Wang, J., Yu, Y.: SeqGAN: Sequence generative adversarial nets with policy gradient. In: Proceedings of the AAAI Conference on Artificial Intelligence (2017). https://doi.org/10.1609/aaai.v31i1.10804

[45] Papagiannis, T., Alexandridis, G., Stafylopatis, A.: Boosting deep reinforcement learning agents with generative data augmentation. Applied Sciences **14**(1), 330 (2024). https://doi.org/10.3390/app14010330

[46] Agarwal, A., Kakade, S., Yang, L.F.: Model-based reinforcement learning with a generative model is minimax optimal. In: Proceedings of the Conference on Learning Theory (2020). https://proceedings.mlr.press/v125/agarwal20b.html

[47] Cao, Y., Sheng, Q.Z., McAuley, J., Yao, L.: Reinforcement learning for generative AI: A survey. arXiv preprint arXiv:2308.14328 (2023). https://arxiv.org/abs/2308.14328

[48] Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., Zaremba, W.: OpenAI gym. arXiv preprint arXiv:1606.01540 (2016). https://arxiv.org/abs/1606.01540