Flow-Based Policy for Online Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present **FlowRL**, a novel framework for online reinforcement learning that integrates flow-based policy representation with Wasserstein-2-regularized optimization. We argue that in addition to training signals, enhancing the expressiveness of the policy class is crucial for the performance gains in RL. Flow-based generative models offer such potential, excelling at capturing complex, multimodal action distributions. However, their direct application in online RL is challenging due to a fundamental objective mismatch: standard flow training optimizes for static data imitation, while RL requires value-based policy optimization through a dynamic buffer, leading to difficult optimization landscapes. FlowRL first models policies via a state-dependent velocity field, generating actions through deterministic ODE integration from noise. We derive a constrained policy search objective that jointly maximizes Q through the flow policy while bounding the Wasserstein-2 distance to a behavior-optimal policy implicitly derived from the replay buffer. This formulation effectively aligns the flow optimization with the RL objective, enabling efficient and value-aware policy learning despite the complexity of the policy class. Empirical evaluations on DMControl and Humanoidbench demonstrate that FlowRL achieves competitive performance in online reinforcement learning benchmarks.

1 Introduction

Recent advances in iterative generative models, particularly Diffusion Models (DM) [16, 37] and Flow Matching (FM) [22, 23, 40], have demonstrated remarkable success in capturing complex multimodal distributions. These models excel in tasks such as high-resolution image synthesis [7], robotic imitation learning [6, 2], and protein structure prediction [17, 3], owing to their expressivity and ability to model stochasticity. A promising yet underexplored application lies in leveraging their multimodal generation capabilities to enhance reinforcement learning (RL) policies, particularly in environments with highly stochastic or multimodal dynamics.

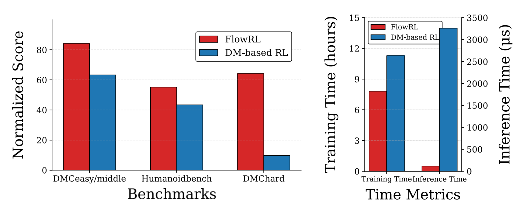


Figure 1: (left) Normalized scores comparing FlowRL and DM-based RL (QVPO) on 12 challenging DMC-hard and HumanoidBench tasks, and 3 DMC-easy & middle tasks. (right) Computational efficiency on the Dogrun task: 1M-step training time and single env step inference time.

Traditional RL frameworks alternate between Q-function estimation and policy updates [38], often parameterizing policies as Gaussian [13] or deterministic policies [36, 12] to maximize expected returns. However, directly employing diffusion or flow-based models as policies introduces a fundamental challenge: the misalignment between RL objectives, which aim to optimize value-aware distributions, and generative modeling, which imitates static data distributions. This discrepancy becomes exacerbated in online RL, where nonstationary data distributions and evolving Q-value estimates lead to unstable training [12].

While recent methods have pioneered the use of diffusion model (DM)-based policies in online reinforcement learning [44, 8, 42], these approaches still suffer from high computational cost and inefficient sample usage (see Section 2). By contrast, flow-based models (FMs), despite their ability to represent complex and multimodal policies, have yet to be effectively integrated into online RL frameworks.

Our method distinguishes itself by leveraging carefully selected replay buffer data as a reference distribution to align flow-based policies with high-value behaviors while preserving multimodality. Inspired by prior works such as SIL [27] and OBAC [25], which utilised behaviour policies to guide policy optimization but limit policy expressivity to capture diverse behaviors, we propose a unified framework that integrates flow-based action generation with Wasserstein-2-regularized [10] distribution matching. Specifically, our policy extraction objective simultaneously maximizes Q-values through flow-based actor and minimizes distribution distance from high-reward trajectories identified in the replay buffer. By reformulating this dual objective as a guided flow-matching loss, we enable the policy to adaptively imitate empirically optimal behaviors while exploring novel actions that maximize future returns. Besides, this approach retains the simplicity of standard actor-critic architectures, without requiring lengthy iterative sampling steps or auxiliary inference tricks [18, 8]—yet fully exploits the multimodality of flow models to discover diverse, high-performing policies. We evaluate our approach on challenging DMControl [39] and HumanoidBench [35], demonstrating competitive performance against state-of-the-art baselines. Notably, our framework achieves one-step policy inference, significantly reducing computational overhead and training instability caused by backpropagation through time (BPTT) [42, 30]. Experimental results highlight both the empirical effectiveness of our method and its practical advantages in scalability and efficiency, establishing a robust pathway for integrating expressive generative models into online RL.

2 Related Work

In this section, we provide a comprehensive survey of existing policy extraction paradigms based on iterative generative models based policy, with a particular focus on recent advances that leverage diffusion and flow-based models in offline or online reinforcement learning. We categorize these approaches according to their underlying policy optimization objective and highlight their respective advantages and limitations.

Generalized Behavior Cloning Generalized Behavior Cloning, often akin to weighted behavioral cloning or weighted regression [32, 31], trains policies by imitating high-reward trajectories from a replay buffer, weighted by advantage or value estimates, thereby avoiding BPTT. Previous methods like EDP [18], QGPO [24], QVPO [8], and QIPO [45] implemented these paradigms, enhancing computational efficiency by bypassing BPTT. However, as demonstrated in prior research, this approach has been empirically shown to be inefficient [29, 30], and often leads to suboptimal performance.

Reverse process as policy parametrizations These methods use reparameterized policy gradients, computing gradients of the Q-function with respect to policy parameters directly through the generative model's reverse sampling process, similar to the reparameterization trick commonly employed in Gaussian-based policies [13]. Previous methods, such as DQL [43], DiffCPS [15], Consistency-AC [9], and DACER [42], backpropagate gradients through the reverse diffusion process, which, while flexible, incurs significant computational costs due to iterative denoising and backpropagation through time (BPTT) [29]. These factors limit the scalability of such algorithms to more complex environments. To address this, FQL [30] distills a one-step policy from a flow-matching policy, reducing computational cost, but requires careful hyperparameter tuning.

Other Approaches. Beyond above methods, alternative methods include action gradients [44, 33], hybrid Markov Decision Processes (MDPs) [34] , rejection sampling [4] or combinations of above strategies [26].

The distinction between these methods underscores an inherent trade-off between computational simplicity and the efficiency of policy extraction. Generalized Behavior Cloning emphasizes ease of implementation, often at the expense of policy extraction efficiency. In contrast, reparameterized policy gradients facilitate direct policy updates but incur increased complexity. These observations highlight the necessity for further research to achieve a better balance between expressivity and scalability when applying iterative generative models to reinforcement learning.

3 Preliminaries

3.1 Reinforcement Learning

Consider the Markov Decision Process (MDPs) [1] defined by a 5-tuple $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma \rangle$, where $\mathcal{S} \in \mathbb{R}^n$ and $\mathcal{A} \in \mathbb{R}^m$ represent the continuous state and action spaces, $\mathcal{P}(s'|s,a) : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ denotes the dynamics distribution of the MDPs, $r(s,a) : \mathcal{S} \times \mathcal{A} \to \Delta(\mathbb{R})$ is a reward function, $\gamma \in [0, 1)$ gives the discounted factor for future rewards. The goal of RL is to find a policy $\pi(a|s) : \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes the cumulative discounted reward:

$$J_\pi = \mathbb{E}_{\pi,\mathcal{P}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right]. \tag{1}$$

In this paper, we focus on the online off-policy RL setting, where the agent interacts with the environment and collects new data into a replay buffer $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a, s', r)\}$. The replay buffer consequently maintains a distribution over trajectories induced by a mixture of historical behavior policies π_β. At the k-th iteration step, the online learning policy is denoted as π_k, with its corresponding Q value function defined by:

$$Q^{\pi_k}(s,a) = \mathbb{E}_{\pi_k,\mathcal{P}} \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a \right], \tag{2}$$

and it can be derived by minimizing the TD error [38]:

$$\begin{aligned} &\arg\min_{Q^{\pi_k}} \; \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\left(Q^{\pi_k}(s,a) - \mathcal{T}^{\pi_k} Q^{\pi_k}(s,a) \right)^2 \right], \\ &\text{where} \quad \mathcal{T}^{\pi_k} Q^{\pi_k}(s,a) = r(s,a) + \gamma \, \mathbb{E}_{s'\sim\mathcal{P}(\cdot|s,a),\, a'\sim\pi_k(\cdot|s')} \left[Q^{\pi_k}(s',a') \right]. \end{aligned} \tag{3}$$

Similarly, we distinguish the following key elements:

- **Optimal policy and Q-function:** The optimal policy π^* maximizes the expected cumulative reward, and the associated Q-function $Q^*(s,a)$ characterizes the highest achievable return.
- **Behavior policy and replay buffer:** The behavior policy π_β is responsible for generating the data stored in the replay buffer [21, 25]. Its Q-function, $Q^{\pi_\beta}(s,a)$, reflects the expected return when following π_β. Notably, $\mathcal{D}$ is closely tied to the distribution of π_β, such that actions sampled from $\mathcal{D}$ are supported by those sampled from π_β (i.e., $a \in \mathcal{D} \Rightarrow a \sim \pi_\beta$).
- **Behavior-optimal policy:** Among all behavior policies present in the buffer, we define π_{β^*} as the one that achieves the highest expected return, with Q-function $Q^{\pi_{\beta^*}}(s,a)$.

These definitions yield the following relationship, which holds for any state-action pair:

$$Q^*(s,a) \geq Q^{\pi_{\beta^*}}(s,a) \geq Q^{\pi_\beta}(s,a). \tag{4}$$

This relationship suggests that, although direct access to the optimal policy is typically infeasible, the value of the optimal behavior policy constitutes a theoretical lower bound [27] on the performance that can be achieved by policies derived from the replay buffer.

3.2 Flow Models

Continuous Normalizing Flows (CNF) [5] model the time-varying probability paths by defining a transformation between an initial distribution p_0 and a target data distribution p_1 [22, 23]. This transformation is parameterized by a flow $\psi_t(x)$ governed by a learned time-dependent vector field $v_t(x)$ [5], following the ordinary differential equation (ODE):

$$\frac{d}{dt}\psi_t(x) = v_t(\psi_t(x)), \tag{5}$$

and the continuity equation [41]:

$$\frac{d}{dt}p_t(x) + \nabla \cdot [p_t(x) v_t(x)] = 0, \quad \forall x \in \mathbb{R}^d. \tag{6}$$

Flow Matching. Flow matching provides a theoretically grounded framework for training continuous-time generative models through deterministic ordinary differential equations (ODEs). Unlike diffusion models that rely on stochastic dynamics governed by stochastic differential equations (SDEs) [37], flow matching operates via a *deterministic* vector field, enabling simpler training objectives and more efficient sampling trajectories. The core objective is to learn a neural velocity field $v_\theta : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ that approximates a predefined conditional target velocity field $u(t, x|x^1)$.

Given a source distribution $q(x^0)$ and target distribution $p(x^1)$, the training process involves minimizing the conditional flow matching objective [22]:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\substack{t\sim\mathcal{U}([0,1]) \\ x^1\sim p,\, x^0\sim q}} \left\| v_\theta\big(t, x^t\big) - u(t, x^t|x^1) \right\|_2^2, \tag{7}$$

where the linear interpolation path is defined as $x^t = tx^1 + (1-t)x^0$ with $u(t, x^t|x^1) = x^1 - x^0$. This formulation induces a *probability flow* governed by the ODE:

$$\frac{dx}{dt} = v_\theta(t, x), \quad x^0 \sim q, \tag{8}$$

which transports samples from q to p.

4 Method

In this section, we detail the design of our method. We first parameterize the policy as a flow model, where actions are generated by integrating a learned velocity field over time. For policy improvement, we model policy learning as a constrained policy search that maximizes expected returns while bounding the distance to an optimal behavior policy. Practically, we circumvent intractable distribution matching and optimal behavior policy by aligning velocity fields with elite historical actions through regularization and implicit guidance, enabling efficient constraint enforcement.

4.1 Flow Model based Policy Representation.

We parameterize π_θ with $v_\theta(t, s, a^t)$, a state-action-time dependent velocity field, as an actor for reinforcement learning. The policy π_θ can be derived by solving ODE (8) :

$$\pi_\theta(s, a^0) = a^0 + \int_0^1 v_\theta(t, s, a^t)dt, \tag{9}$$

where $a^0 \sim \mathcal{N}(0, I^2)$. The superscript t denotes the continuous time variable in the flow-based ODE process to distinguish it from discrete Markovian time steps in reinforcement learning. (For brevity, the terminal condition at $t = 1$ is omitted in the notation.) The Flow Model derives a deterministic velocity field v_θ from an ordinary differential equation (ODE). However, when a^0 is sampled from a random distribution, the model effectively functions as a stochastic actor, exhibiting diverse behaviors across sampling instances. This diversity in generated trajectories inherently promotes enhanced exploration in online reinforcement learning.

Recall the definition in Section 3.1. Following the notation of π_β and π_{β^*}, we can define the corresponding velocity fields as follows:

Let v_β be the velocity field induced by the behavior policy π_β, such that:

$$v_\beta(s, a) = a - a^0.$$

where $s, a \sim \mathcal{D}$, and $a^0 \sim \mathcal{N}(0, I^2)$.

Similarly, let v_{β^*} denote the velocity field induced by the behavior-optimal policy π_{β^*}:

$$v_{\beta^*}(s, a) = a - a^0.$$

where $a \sim \pi_{\beta^*}$, and $a^0 \sim \mathcal{N}(0, I^2)$.

4.2 Optimal-Behavior Constrained Policy Search with Flow Models

Building on the discussion in Section 3.1, where the optimal behavior policy is established as a lower bound for the optimal policy, we proceed to optimize the following objective under a constrained policy search setting:

$$\begin{aligned} \theta^* = \ & \arg\max_{\theta} \ \ \mathbb{E}_{a\sim\pi_\theta}\left[Q^{\pi_\theta}(s,a)\right], \\ & \text{s.t.} \ \ D\left(\pi_\theta, \pi_{\beta^*}\right) \leq \epsilon. \end{aligned} \tag{10}$$

Here, $D(\pi_\theta, \pi_{\beta^*})$ denotes a distance metric between the current policy and the optimal behavior policy distributions.

The objective is to maximize the expected reward $\mathbb{E}_{a\sim\pi_\theta}[Q^\pi(s,a)]$ while constraining the learned policy π_θ to remain within an ϵ-neighborhood of the optimal behavior policy π_{β^*}, i.e., $D(\pi_\theta, \pi_{\beta^*}) \leq \epsilon$. This formulation utilizes the Q-function, a widely used and effective approach for policy extraction, while ensuring fidelity to the optimal behavior policy. Despite its theoretical appeal, this optimization paradigm exhibits two inherent limitations:

- *Challenges in computing distributional distances*: For flow-based models, computing policy densities at arbitrary samples is computationally expensive, which limits the practicality of distance metrics such as the KL divergence for sample-based estimation and policy regularization.
- *Inaccessibility of the optimal behavior policy* π_{β^*}: The replay buffer contains trajectories from a mixture of policies, making it difficult to directly sample from π_{β^*} or to reliably estimate its associated velocity field, thereby complicating the computation of related quantities in practice.

4.3 A Tractable Surrogate Objective

To overcome the aforementioned challenges, we propose the following solutions:

- **Wasserstein Distance as Policy Constraints:** We introduce a policy regularization method based on the alignment of velocity fields. This approach bounds the Wasserstein distance between policies by characterizing their induced dynamic transport processes, thereby imposing direct empirical constraints on the evolution of policies without requiring density estimation.
- **Implicit Guidance for Optimal Behaviors:** Instead of explicitly constraining the policy to match the inaccessible π_{β^*}, we leverage implicit guidance from past best-performing behaviors in the buffer, enabling efficient revisiting of arbitrary samples and encouraging the policy to remain within a high-quality region of the action space.

In particular, we adopt the squared Wasserstein-2 distance for its convexity with respect to the policy distribution and ease of implementation. This metric is also well-suited for measuring the velocity field between policies and enables efficient sample-based regularization within the flow-based modeling framework. In general, we can define the Wasserstein-2 Distance [41] as follows :

Definition 4.1 (Wasserstein-2 Distance) Given two probability measures p and q on $\mathbb{R}^n$, the squared Wasserstein-2 distance between p and q is defined as:

$$W_2^2(p,q) = \inf_{\gamma\in\Pi(p,q)} \int_{\mathbb{R}^n\times\mathbb{R}^n} \gamma(x,y)\|x-y\|^2 dxdy, \tag{11}$$

where $\Pi(p,q)$ denotes the joint distributions of p and q, γ on $\mathbb{R}^n \times \mathbb{R}^n$ with marginals p and q. Specifically, we derive a tractable upper bound for the Wasserstein-2 distance (proof in A.1):

Theorem 4.1 (W-2 Bound for Flow Matching) Let v_θ and v_{β^*} be two velocity fields inducing time-evolving distributions $\pi_\theta^t(a|s)$ and $\pi_{\beta^*}^t(a|s)$, respectively. Assume v_β is Lipschitz continuous in a with constant L. $a^t = ta + (1-t)a^0$. Then, the squared Wasserstein-2 distance between π_θ and π_{β^*} at $t=1$ satisfies:

$$W_2^2\left(\pi_\theta, \pi_{\beta^*}\right) \leq e^{2L}\int_0^1 \mathbb{E}_{a\sim\pi_{\beta^*}}\left[\left\|v_\theta(s,a^t,t) - v_{\beta^*}\right\|^2\right]dt. \tag{12}$$

By explicitly constraining the Wasserstein-2 distance, the model enforces proximity between the current policy and the optimal policy stored in the buffer. This objective is inherently consistent with the generative modeling goal of minimizing distributional divergence. The regularization mechanism benefits from the representational expressiveness of flow-based models in capturing diverse, high-performing action distributions while systematically restricting policy updates.

However, while the upper bound of Wasserstein-2 distance above is theoretically tractable, sampling directly from π_{β^*} or evaluating its velocity field remains a computational barrier in practice. To circumvent this limitation, we introduce an implicit guidance (13) mechanism through the $Q^{\pi_{\beta^*}}$, which is more readily estimable:

$$\mathbb{E}_{\substack{a' \sim \pi_\theta, t \sim \mathcal{U}(0,1) \\ s,a \sim \mathcal{D}}} \left[f\left(Q^{\pi_{\beta^*}}(s,a) - Q^{\pi_\theta}(s,a')\right) \left\| v_\theta(s, a^t, t) - (a - a^0) \right\|^2 \right], \tag{13}$$

$$f \propto \max\left(Q^{\pi_{\beta^*}} - Q^{\pi_\theta}, 0\right). \tag{14}$$

The constraint incorporates a non-negative weighting function, as defined in Eq. (14), thereby establishing an adaptive regularization mechanism. A positive value of f signifies that the behavioral policy achieves superior performance relative to the current policy; under these circumstances, the constraint adaptively regularizes the current policy towards the optimal behavioral policy.

The implicit form of the constraints in Eq. (12) enables efficient utilization of arbitrary samples from the replay buffer, thus improving sample efficiency. Moreover, by relaxing the strict constraint on the Wasserstein-2 distance, the modified objective enhances computational efficiency. Notwithstanding this relaxation, policy improvement guarantees remain valid, as demonstrated in the following theorem (proof in Appendix A.2):

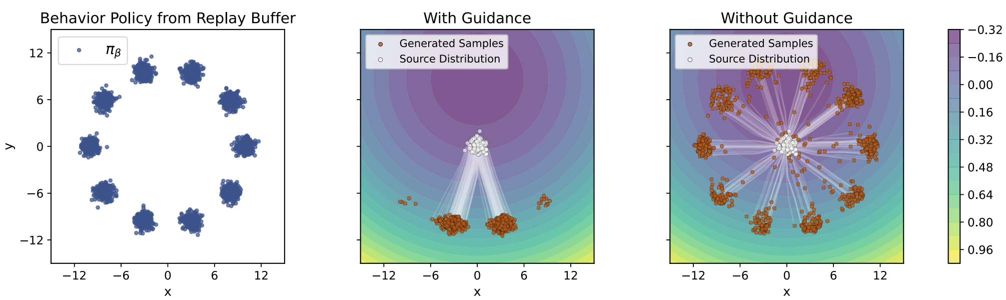


Figure 2: Illustration of Theorem 4.2 on a bandit toy example: (left) behavior data in the replay buffer; (middle) implicit value-guided flow matching steers the policy toward the high-performance behavior policy(π_{β^*}), heatmap shows $Q^{\pi_k} - Q^{\pi_{\beta^*}}$, white lines indicate transport paths; (right) standard flow matching leads to dispersed sampling with high variance under limited flow steps.

Theorem 4.2 (Weighted CFM) Let $\pi_k(a|s)$ be the current policy induced by velocity field v_{θ_k}, and f, a non-negative weighting function with $f \propto Q^{\pi_{\beta^*}} - Q^{\pi_k}$. Minimizing the objective (13) yields an improved policy distribution:

$$\pi_{k+1}(a|s) = \frac{f(s,a)\pi_{\beta^*}(a|s)}{\mathcal{Z}(s)}, \tag{15}$$

where $\mathcal{Z}(s) = \int_{\mathcal{A}} f \cdot \pi_k(a|s)\, da$ is the normalization factor.

Figure 2 shows that, as guaranteed by Theorem 4.2, flow matching with guidance can steer the policy toward the π_{β^*}, even without direct sampling from it. For details of the toy example settings, see Appendix B.4.

4.4 A Practical Implementation

Building on the theoretical developments above, we now present a practical implementation of FlowRL, as detailed in Algorithm 1.

Policy Evaluation Recall the constraint in Eq. (13), which necessitates the evaluation of both the current policy value function Q^{π_θ} and the optimal behavioral policy value function $Q^{\pi_{\beta^*}}$. The value function Q^{π_θ} is estimated using standard Bellman residual minimization, as described in Eq. (3). For $Q^{\pi_{\beta^*}}$, leveraging the definition of π_{β^*}, we similarly adopt the following objective:

$$\arg\min_{Q^{\pi_{\beta^*}}} \ \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\left(Q^{\pi_{\beta^*}}(s,a) - \mathcal{T}^{\pi_{\beta^*}} Q^{\pi_{\beta^*}}(s,a) \right)^2 \right], \tag{16}$$

$$\mathcal{T}^{\pi_{\beta^*}} Q^{\pi_{\beta^*}}(s,a) = r(s,a) + \gamma\, \mathbb{E}_{s'\sim\mathcal{D}} \left[\max_{a'\sim\mathcal{D}} Q^{\pi_{\beta^*}}(s',a') \right]. \tag{17}$$

To circumvent the difficulties of directly evaluating the max operator, we leverage techniques from offline reinforcement learning to estimate $Q^{\pi_{\beta^*}}$. Among these approaches, we adopt expectile regression [19] due to its simplicity and compatibility with unmodified data pipelines. Specifically, the value function $V^{\pi_{\beta^*}}$ and the action-value function $Q^{\pi_{\beta^*}}$ are estimated by solving the following optimization problems:

$$\arg\min_{V^{\pi_{\beta^*}}} \mathbb{E}_{(s,a)\sim\mathcal{D}} \left[L_2^\tau \left(Q^{\pi_{\beta^*}}(s,a) - V^{\pi_{\beta^*}}(s) \right) \right], \tag{18}$$

$$\arg\min_{Q^{\pi_{\beta^*}}} \mathbb{E}_{(s,a,s',r)\sim\mathcal{D}} \left[\left(r + \gamma V^{\pi_{\beta^*}}(s') - Q^{\pi_{\beta^*}}(s,a) \right)^2 \right], \tag{19}$$

where $L_2^\tau(x) = |\tau - \mathbb{1}(x<0)|x^2$ denotes the expectile regression loss and τ is the expectile factor.

Policy Extraction Accordingly, the policy extraction problem for flow-based models can be formulated as the following constrained optimization:

$$\theta^* = \arg\max_\theta \ \ \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi_\theta} \left[Q^{\pi_\theta}(s,a) \right], \tag{20}$$

$$\text{s.t. } \ \mathbb{E}_{s,a\sim\mathcal{D},a'\sim\pi_\theta} \left[f\left(Q^{\pi_{\beta^*}} - Q^{\pi_\theta} \right) \left\| v_\theta(s,a^t,t) - (a - a^0) \right\|^2 \right] \le \epsilon. \tag{21}$$

Although a closed-form solution can be derived using the Lagrangian multiplier and KKT conditions, it is generally intractable to apply in practice due to the unknown partition function [31, 32, 25]. Therefore, we adopt a Lagrangian form, leading to the following objective:

$$\mathcal{L}(\theta) = \mathbb{E}_{s,a\sim\mathcal{D},a'\sim\pi_\theta} [\underbrace{Q^{\pi_\theta}(s,a')}_{\text{exploration}} - \lambda \left(\underbrace{f(Q^{\pi_{\beta^*}} - Q^{\pi_\theta}) \| v_\theta - (a - a^0) \|^2}_{\text{exploitation}} - \epsilon \right)]. \tag{22}$$

Where λ is the Lagrangian multiplier, which is often set as a constant in practice [11, 20].

Objective (22) can be interpreted as comprising two key components: (1) maximization of the learned Q-function, which encourages the agent to explore unknown regions and facilitates policy improvement; and (2) a policy distribution regularization term, which enforces alignment with optimal behavior policies and thereby promotes the exploitation of high-quality actions.

Conceptual similarities exist between our method and both self-imitation learning [27] and tandem learning [28]. Self-imitation learning focuses on exploiting high-reward behaviors by encouraging the policy to revisit successful past experiences, typically requiring complete trajectories and modifications to the data pipeline. In contrast, our method operates directly on individual samples from the buffer, enabling more flexible and efficient sample utilization. Tandem learning, by comparison, decomposes the learning process into active and passive agents to facilitate knowledge transfer, with a primary emphasis on value learning, whereas our approach is centered on policy extraction.

5 Experiments

To comprehensively evaluate the effectiveness and generality of **FlowRL**, we conduct experiments on a diverse set of challenging tasks from DMControl [39] and HumanoidBench [35]. These benchmarks encompass high-dimensional locomotion and human-like robot (Unitree H1) control tasks. Our evaluation aims to answer the following key questions:

Algorithm 1 Flow RL

Require: Critic Q^{π_θ}, critic $Q^{\pi^*_\beta}$, value $V^{\pi^*_\beta}$, flow model v_θ, replay buffer $\mathcal{D} = \emptyset$, weighting function f

repeat
  for each environment step **do**
    $a \sim \pi_\theta(a|s), \quad r, s' \sim P(s'|s,a)$
    $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s,a,s',r)\}$
  end for
  for each gradient step **do**
    Estimate value for π_θ : Update Q^{π_θ} by (3),
    Estimate value for π_{β^*} : Update $Q^{\pi_{\beta^*}}$ by (19), update $V^{\pi_{\beta^*}}$ by (18)
    Update v_θ by (22)
  end for
until reach the max environment steps

1. How does FlowRL compare to previous online RL algorithms and existing diffusion-based online algorithms?
2. Can the algorithm still demonstrate strong performance in the absence of any explicit exploration mechanism?
3. How does the constraint affect the performance?

We compare **FlowRL** against two categories of baselines to ensure comprehensive evaluation: **(1) Model-free RL:** We consider three representative policy parameterizations: deterministic policies (TD3 [12]), Gaussian policies (SAC [13]), and diffusion-based policies (QVPO [8], the previous state-of-the-art for diffusion-based online RL). **(2) Model-based RL:** TD-MPC2 [14], a strong model-based method on these benchmarks, is included for reference only, as it is not directly comparable to model-free methods.

5.1 Results and Analysis

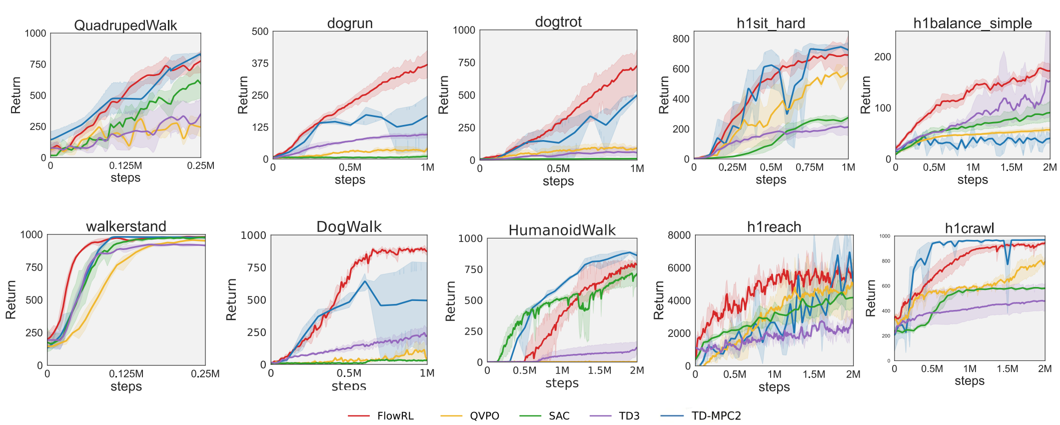
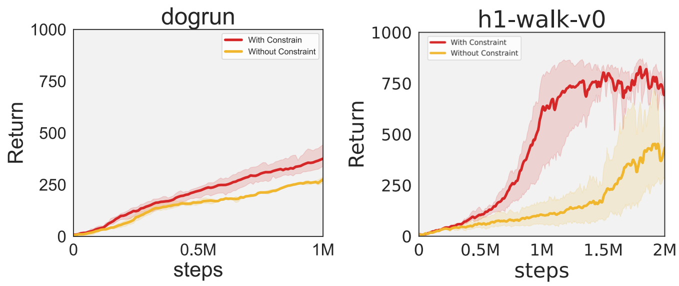
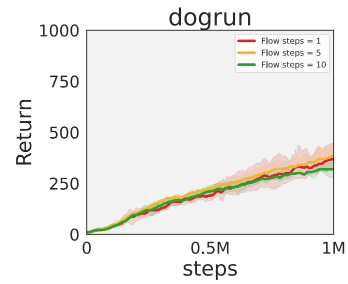


Figure 3: Main results. We provide performance comparisons for tasks (first column: DMC-easy/middle; second and third columns: DMC-hard; fourth and fifth columns: HumanoidBench). For comprehensive results, please refer to Appendix D. All model-free algorithms (FlowRL, SAC, QVPO, TD3) are evaluated with 5 random seeds, while the model-based algorithm (TD-MPC2) uses 3 seeds. Note that direct comparison between model-free methods and the model-based TD-MPC2 is not strictly fair; TD-MPC2 is included just as a reference.

The main results are summarized in Figure 3, which shows the learning curves across tasks. **FlowRL** consistently outperforms or matches the model-free baselines on the majority of tasks, demonstrating strong generalization and robustness, especially in challenging high-dimensional (e.g., the DMC dog domain, where $s \in \mathbb{R}^{223}$ and $a \in \mathbb{R}^{38}$) and complex control settings (e.g., Unitree H1). Compared



(a) Effect of the constraint: FlowRL with theconstraint achieves higher returns compared to the variant without the constraint.



(b) Sensitivity to flow steps: The number of flow steps has a limited effect on FlowRL performance.

Figure 4: Ablation studies

to strong model-based baselines, FlowRL achieves comparable results but is much more efficient in terms of wall-clock time. Notably, both during the training and evaluation stage, we use flow steps $N = 1$, and do not employ any sampling-based action selection used in [8, 18]. Despite the absence of any explicit exploration mechanism, FlowRL demonstrates strong results, which can be attributed to both the inherent stochasticity and exploratory capacity of the flow-based actor and the effective exploitation of advantageous actions identified by the policy constraint. These findings indicate that, while exploration facilitates the discovery of high-reward actions, the exploitation of previously identified advantageous behaviors is equally essential.

5.2 Ablation Studies

One of the central designs in FlowRL is the introduction of a policy constraint mechanism. This design aims to guide the policy towards optimal behavior by adaptively weighting the constraint based on the relative advantage of the optimal behavioral policy over the current policy. To rigorously assess the necessity and effectiveness of this component, we address **Q3** by conducting ablation studies in which the policy constraint is omitted from FlowRL. Experimental results in Figure 4a indicate that the presence of the policy constraint leads to improvements in performance and, by constraining the current policy towards the optimal behavioral policy, enhances sample efficiency. These benefits are especially pronounced in environments with complex dynamics (e.g., H1 control tasks from HumanoidBench), highlighting the importance of adaptive policy regularization in challenging task settings.

We also investigate the sensitivity of the algorithm to different choices of the number of flow steps (N=1,5,10). Experimental results in Figure 4b demonstrate that varying the number of flow steps has only a limited impact on the overall performance. Specifically, using a smaller number of flow steps does not substantially affect the final policy performance. On the other hand, increasing the number of flow steps results in longer backpropagation through time (BPTT) chains, which significantly increases computational complexity and training time. These findings suggest that FlowRL is robust to the choice of flow step and that single-step inference is generally sufficient for achieving stable and efficient learning in practice.

6 Conclusion

We introduces FlowRL, a practical framework that integrates flow-based generative models into online reinforcement learning through Wasserstein-2 distance constrained policy search. By parameterizing policies as state-dependent velocity fields, FlowRL leverages the expressivity of flow models to model action distributions. To align policy updates with value maximization, we propose an implicit guidance mechanism that regularizes the learned policy using high-performing actions from the replay buffer. This approach avoids explicit density estimation and reduces iterative sampling steps, achieving stable training and improved sample efficiency. Empirical results demonstrate that FlowRL achieves competitive performance.

References

[1] Richard Bellman. A markovian decision process. *Journal of mathematics and mechanics*, pages 679–684, 1957.

[2] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. A vision-languageaction flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2(3):5, 2024.

[3] Avishek Joey Bose, Tara Akhound-Sadegh, Guillaume Huguet, Kilian Fatras, Jarrid Rector-Brooks, Cheng-Hao Liu, Andrei Cristian Nica, Maksym Korablyov, Michael Bronstein, and Alexander Tong. Se (3)-stochastic flow matching for protein backbone generation. *arXiv preprint arXiv:2310.02391*, 2023.

[4] Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. *arXiv preprint arXiv:2209.14548*, 2022.

[5] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[6] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[7] Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

[8] Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. *arXiv preprint arXiv:2405.16173*, 2024.

[9] Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. *arXiv preprint arXiv:2309.16984*, 2023.

[10] Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with wasserstein regularization. *arXiv preprint arXiv:2502.06061*, 2025.

[11] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

[12] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

[13] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

[14] Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[15] Longxiang He, Li Shen, Linrui Zhang, Junbo Tan, and Xueqian Wang. Diffcps: Diffusion model based constrained policy search for offline reinforcement learning. *arXiv preprint arXiv:2310.05333*, 2023.

[16] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[17] Bowen Jing, Bonnie Berger, and Tommi Jaakkola. Alphafold meets flow matching for generating protein ensembles. *arXiv preprint arXiv:2402.04845*, 2024.

[18] Bingyi Kang, Xiao Ma, Chao Du, Tianyu Pang, and Shuicheng Yan. Efficient diffusion policies for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36: 67195–67212, 2023.

[19] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

[20] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

[21] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[22] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[23] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[24] Cheng Lu, Huayu Chen, Jianfei Chen, Hang Su, Chongxuan Li, and Jun Zhu. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *International Conference on Machine Learning*, pages 22825–22855. PMLR, 2023.

[25] Yu Luo, Tianying Ji, Fuchun Sun, Jianwei Zhang, Huazhe Xu, and Xianyuan Zhan. Offline-boosted actor-critic: Adaptively blending optimal historical behaviors in deep off-policy rl. *arXiv preprint arXiv:2405.18520*, 2024.

[26] Liyuan Mao, Haoran Xu, Xianyuan Zhan, Weinan Zhang, and Amy Zhang. Diffusion-dice: In-sample diffusion guidance for offline reinforcement learning. *arXiv preprint arXiv:2407.20109*, 2024.

[27] Junhyuk Oh, Yijie Guo, Satinder Singh, and Honglak Lee. Self-imitation learning. In *International conference on machine learning*, pages 3878–3887. PMLR, 2018.

[28] Georg Ostrovski, Pablo Samuel Castro, and Will Dabney. The difficulty of passive learning in deep reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 23283–23295, 2021.

[29] Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? *arXiv preprint arXiv:2406.09329*, 2024.

[30] Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025.

[31] Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

[32] Jan Peters and Stefan Schaal. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th international conference on Machine learning*, pages 745–750, 2007.

[33] Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via q-score matching. *arXiv preprint arXiv:2312.11752*, 2023.

[34] Allen Z Ren, Justin Lidard, Lars L Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. *arXiv preprint arXiv:2409.00588*, 2024.

[35] Carmelo Sferrazza, Dun-Ming Huang, Xingyu Lin, Youngwoon Lee, and Pieter Abbeel. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. *arXiv preprint arXiv:2403.10506*, 2024.

[36] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pages 387–395. Pmlr, 2014.

[37] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[38] Richard S Sutton, Andrew G Barto, et al. Reinforcement learning. *Journal of Cognitive Neuroscience*, 11(1):126–134, 1999.

[39] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[40] Alexander Tong, Kilian Fatras, Nikolay Malkin, Guillaume Huguet, Yanlei Zhang, Jarrid Rector-Brooks, Guy Wolf, and Yoshua Bengio. Improving and generalizing flow-based generative models with minibatch optimal transport. *arXiv preprint arXiv:2302.00482*, 2023.

[41] Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2008.

[42] Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, et al. Diffusion actor-critic with entropy regulator. *Advances in Neural Information Processing Systems*, 37:54183–54204, 2024.

[43] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[44] Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

[45] Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. *arXiv preprint arXiv:2503.04975*, 2025.

A Proofs in the Main Text

Here, we present a sketch of theoretical analyses in Figure 5. We model the policy learning as a constrained policy search that maximizes expected returns while bounding the distance to an optimal behavior policy. To avoid sampling from π_β^*, we employ guided flow matching, which allows the constraint to utilize arbitrary data from the buffer. Finally, we solve the problem using Lagrangian relaxation.

A.1 Proof for Theorem 4.1

Before the proof, we first introduce the following lemma [10]:

Lemma 1 :Let $\psi_1^t(x_0)$ and $\psi_2^t(x_0)$ be the two different flow maps induced by v_1^t and v_2^t starting from x^0, and assume v_2^t are Lipschitz continuous in x with constant L. Define their difference as $\Delta_t(x^0) = \psi_1^t(x^0) - \psi_2^t(x^1)$. (For notational consistency, we denote the time variable as a superscript.) Then the difference satisfies the following inequality:

$$\frac{d}{dt}\Delta_t(x_0) \leq ||v_1^t(\psi_1^t(x^0)) - v_2^t(\psi_1^t(x^0))|| + L||\Delta_t(x_0)||$$

By rewriting equivalently, we have:

$$\frac{d}{dt}\Delta_t(x^0) = \underbrace{v_1^t(\psi_1^t(x^0)) - v_2^t(\psi_1^t(x^0))}_{\delta_v(t)} + \underbrace{v_2^t(\psi_1^t(x^1)) - v_2^t(\psi_2^t(x^0))}_{\delta_\psi(t)}$$

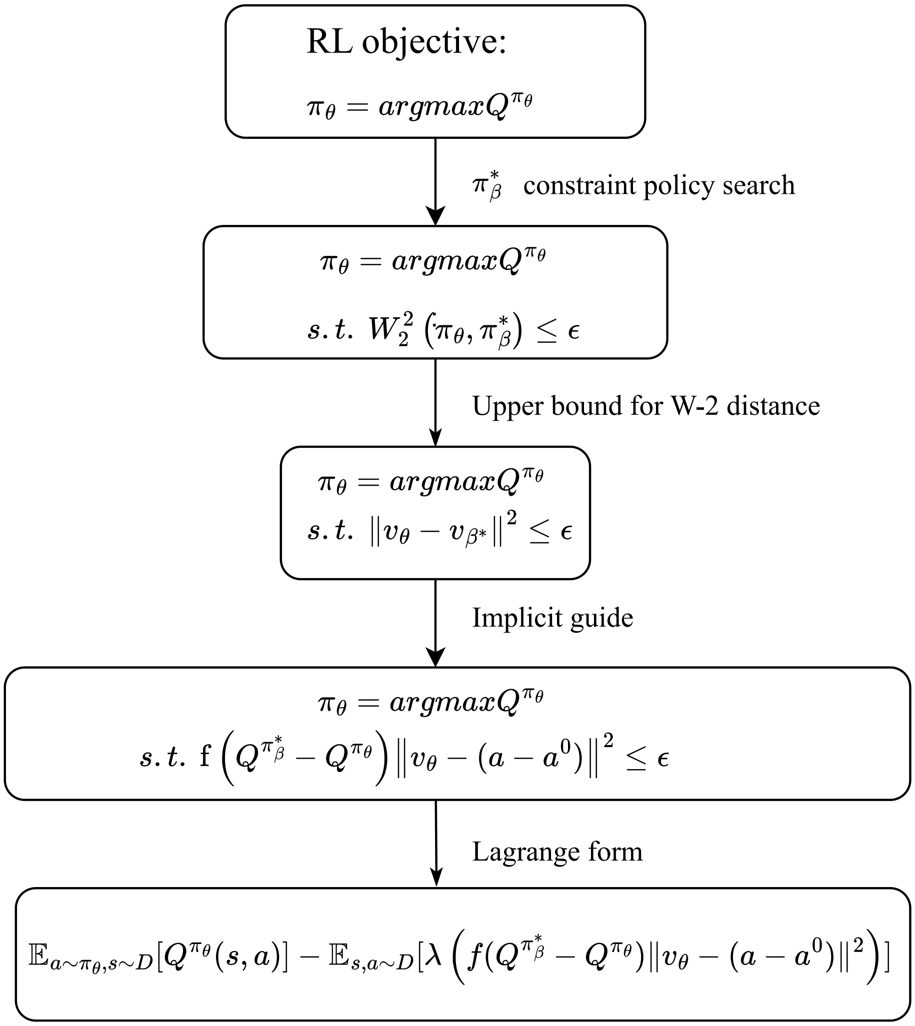


Figure 5: Theoretical sketch of FlowRL

Since v_2^t is Lipschitz continuous in x with constant L, we have:

$$\|v_2^t(x) - v_2^t(y)\| \leq L\|x - y\|$$

By Lipschitz continuity,

$$\|\delta_\psi(t)\| \leq L\|\Delta_t(x^0)\|$$

Then,

$$\left\|\frac{d}{dt}\Delta_t(x^0)\right\| \leq \|\delta_v(t)\| + L\|\Delta_t(x^0)\|$$

This concludes the proof of the inequality satisfied by the difference of the two flow maps.

Let v_θ and v_{β^*} be two velocity fields that induce time-evolving distributions $\pi_\theta^t(a|s)$ and $\pi_{\beta^*}^t(a|s)$, respectively((we omit the superscript $t = 1$ for policy distributions, i.e., $\pi_\theta(a|s) := \pi_\theta^1(a|s)$).). Assume v_{β^*} is Lipschitz continuous with constant L. Then, define $f(t) = \|\Delta_t(x_0)\|$, by Lemma 1,we have:

$$\frac{d}{dt}f(t) \leq \|\delta_v(t)\| + Lf(t),$$

where $\delta_v(t) = v_\theta(t, \psi_t^\theta(s, a^0)) - v_{\beta^*}$. Then, we have,

$$\frac{d}{dt}\left(e^{-Lt} f(t)\right) \le e^{-Lt}\|\delta_v(t)\|.$$

Then we can get (by simply intergrating from 0 to t both side and multiplying e^{-Lt}):

$$e^{-Lt} f(t) - f(0) \le \int_0^t e^{-Lm}\|\delta_v(m)\| dm.$$

The initial policy distribution $a^0 \sim p(a^0)$ is shared between the two velocity fields, so $f(0) = 0$. Therefore,

$$f(t) \le e^{Lt}\int_0^t e^{-Lm}\|\delta_v(m)\| dm.$$

At $t = 1$,

$$f(1) \le e^{L}\int_0^1 e^{-Lm}\|v_\theta(s, \psi_\theta^t(s, a^0), m) - v_{\beta^*}\| dm.$$

By taking the expectation and using Jensen's inequality:

$$\mathbb{E}_{a^0}[f(1)^2] \le e^{2L}\int_0^1 \mathbb{E}_{a\sim\pi_\theta^t}[\|v_\theta(s, a, t) - v_{\beta^*}\|^2] dt.$$

And use the definition of the Wasserstein-2 distance:

$$W_2^2(\pi_\theta, \pi_{\beta^*}) = \inf_{\gamma\in\Pi(\pi_\theta, \pi_{\beta^*})}\int_{\mathbb{R}^n\times\mathbb{R}^n}\|x - y\|^2 d\gamma(x, y),$$

where $\Pi(\pi_\theta, \pi_{\beta^*})$ denotes the set of all couplings between π_θ and π_{β^*}. Construct the following coupling γ and define:

- $a_\theta^1 = \psi_\theta^1(x_0)$,
- $a_{\beta^*}^1 = \psi_{\beta^*}^1(x_0)$.

By definition, the coupling γ is defined via the joint distribution of $(a \sim \pi_\theta, a \sim \pi_{\beta^*})$ induced by $a_0 \sim p_0$. So, for any coupling γ,

$$W_2^2(\pi_\theta, \pi_{\beta^*}) \le \int_{\mathbb{R}^n\times\mathbb{R}^n}\|x - y\|^2 d\gamma(x, y).$$

With the constructed coupling substituted, we have

$$\int_{\mathbb{R}^n\times\mathbb{R}^n}\|x - y\|^2 d\gamma(x, y) = \mathbb{E}_{a^0}\left[\|\psi_\theta^1(a^0) - \psi_{\beta^*}^1(a^0)\|^2\right] = \mathbb{E}_{a^0}[f(1)^2].$$

Recall that the flow-based policy models transport the initial distribution $p_0(a^0)$ to the final policy distributions π_θ and π_{β^*} at $t = 1$. The squared Wasserstein-2 distance between π_θ and π_{β^*} can be bounded as

$$W_2^2(\pi_\theta, \pi_{\beta^*}) \le \mathbb{E}_{a^0}[f(1)^2]. \tag{23}$$

Thus,

$$W_2^2(\pi_\theta, \pi_{\beta^*}) \le e^{2L}\int_0^1 \mathbb{E}_{a\sim\pi_\theta^t}[\|v_\theta(s, a^t) - v_{\beta^*}(s, a)\|^2] ds. \tag{24}$$

A.2 Proof for Theorem 4.2

The weighted loss can be written as:

$$\mathcal{L}_{\mathrm{W}}(\theta) = \int_{s\sim D}\rho(s)\int_{s,a\sim D} f(s, a)\,\pi_k(a|s)\,\|v_\theta(s, a^t, t) - (a - a^0))\| da\, ds$$

where $\rho(s)$ is the state distribution in replay buffer, $a^0 \sim \mathcal{N}(0, I^2)$, $t \sim \mathcal{U}(0, 1)$, $a^t = ta + (1-t)a^0$.

Assuming the weighted policy distribution is:

$$\pi_{k+1}(a'|s) = \frac{f(s,a)\,\pi_k(a|s)}{\mathcal{Z}(s)}, \quad \text{where} \quad \mathcal{Z}(s) = \int_{s,a\sim D} f(s,a)\,\pi_k(a|s)\,da.$$

Substituting above $\pi_{k+1}(a'|s)$ into the loss function, we have:

$$\mathcal{L}_{\mathrm{W}}(\theta) = \int_{s\sim D} \rho(s)\,\mathcal{Z}(s) \int_{s,a\sim D} \pi_{k+1}(a'|s)\,\|v_\theta(s,a^t,t) - (a - a^0))\|da\,ds.$$

The expectation form:

$$\mathcal{L}_{\mathrm{W}}(\theta) = \mathbb{E}_{s\sim\mathcal{D},\,a\sim\pi_{k+1}(a|s)}\left[\mathcal{Z}(s)\,\|v_\theta(s,a^t,t) - (a - a^0))\|\right].$$

The gradient of $\mathcal{L}_{\mathrm{W}}(\theta)$ is:

$$\nabla_\theta \mathcal{L}_{\mathrm{W}}(\theta) = \mathbb{E}_{s\sim\mathcal{D},\,a\sim\pi_{k+1}(a|s)}\left[\mathcal{Z}(s)\,\nabla_\theta\|v_\theta(s,a,t) - (a - a^0))\|\right].$$

$\mathcal{Z}(s)$ does not depend on θ, that means, minimizing $\mathcal{L}_{\mathrm{W}}(\theta)$ is equivalent to minimizing the expected loss under the new distribution $\pi_{k+1}(a|s)$, provided that our assumption holds.

B Hyperparameters and Experiment Settings

In this section, we provide comprehensive details regarding the implementation of FlowRL, the baseline algorithms, and the experimental environments. All experiments are conducted on a single NVIDIA H100 GPU and an Intel(R) Platinum 8480C CPU, with two tasks running in parallel on the GPU.

B.1 Hyperparameters

The hyperparameters used in our experiments are summarized in Table 1. For the choice of the weighting function, we use $f(x) = \mathbb{I}(x)\cdot\exp(x)$, where $\mathbb{I}(x)$ is the indicator function, i.e.,

$$\mathbb{I}(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{otherwise} \end{cases}$$

For numerical stability, the Q function is normalized by subtracting its mean exclusively during the computation of the weighting function.

B.2 Baselines

In our experiments, we have implemented SAC, TD3, QVPO and TD-MPC2 using their original code bases and slightly tuned them to match our evaluation protocol to ensure a fair and consistent comparison.

- For SAC [13], we utilized the open-source PyTorch implementation, available at `https://github.com/pranz24/pytorch-soft-actor-critic`.
- TD3 [12] was integrated into our experiments through its official codebase, accessible at `https://github.com/sfujim/TD3`.
- QVPO [8] was integrated into our experiments through its official codebase, accessible at `https://https://github.com/wadx2019/qvpo`.
- TD-MPC2 [14] was employed with its official implementation from `https://github.com/nicklashansen/tdmpc2` and used their official results.

B.3 Environment Details

We validate our algorithm on the DMControl [39] and HumanoidBench [35], including the most challenging high-dimensional and Unitree H1 humanoid robot control tasks. On DMControl, tasks are categorized into DMC easy & middle (walker and quadruped domains), and DMC hard (dog and humanoid domains). On HumanoidBench, we focus on tasks that do not require dexterous hands.

Table 1: Hyperparameters

	Hyperparameter	Value
Hyperparameters	Optimizer	Adam
	Critic learning rate	3×10^{-4}
	Actor learning rate	3×10^{-4}
	Discount factor	0.99
	Batchsize	256
	Replay buffer size	1×10^{6}
	Expectile factor τ	0.9
	Lagrangian multiplier λ	0.1
	Flow steps N	1
	ODE Slover	Midpoint Euler
Value network	Network hidden dim	512
	Network hidden layers	3
	Network activation function	mish
Policy network	Network hidden dim	512
	Network hidden layers	2
	Network activation function	elu

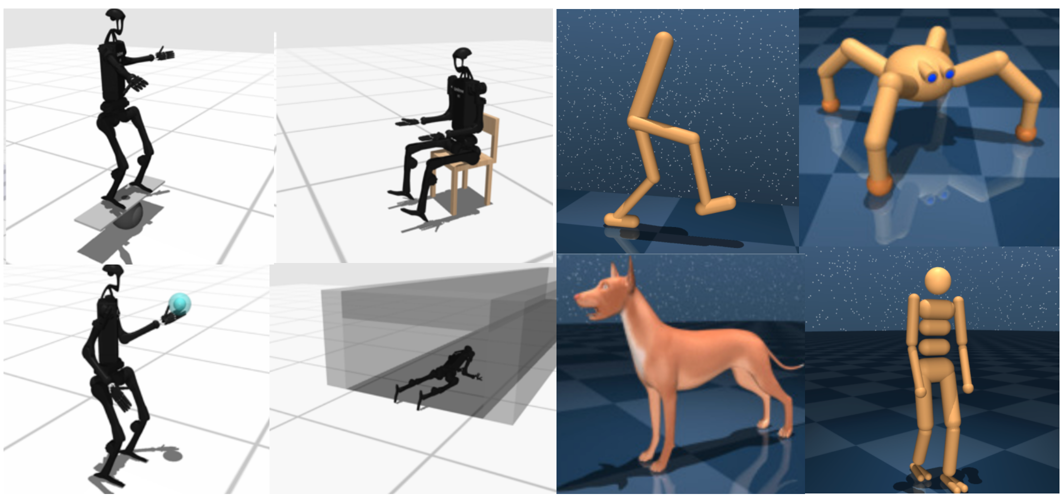


Figure 6: Task domain visualizations

B.4 Toy Example Setup

We consider a 2D toy example as follows. The behavior policy is a Gaussian mixture model with 10 components, each with mean

$$\mu_k = (10\cos(2\pi k/10),\ 10\sin(2\pi k/10)), \quad k = 0, 1, \ldots, 9,$$

and covariance I. The initial distribution is a Gaussian $\mathcal{N}((0,0),\ I)$. $Q^{\pi_{\beta^*}} - Q^{\pi_\theta}$ is defined as

$$\frac{1}{600}\|x - (0,\ 8.66)\|^2 - 3,$$

and $f(x) = \mathbb{I}(x) \cdot x$. Flow steps $N = 5$.

C Limitation and Future Work

In this work, we propose a flow-based reinforcement learning framework that leverages the behavior-optimal policy as a constraint. Although competitive performance is achieved even without explicit exploration, investigating efficient adaptive exploration mechanisms remains a promising direction for future research.

Task	State dim	Action dim
Walker Run	24	6
Walker Stand	24	6
Quadruped Walk	78	12
Humanoid Run	67	24
Humanoid Walk	67	24
Dog Run	223	38
Dog Trot	223	38
Dog Stand	223	38
Dog Walk	223	38

Table 2: Task dimensions for DMControl.

Task	Observation dim	Action dim
H1 Balance Hard	77	19
H1 Balance Simple	64	19
H1 Crawl	51	19
H1 Maze	51	19
H1 Reach	57	19
H1 Sit Hard	64	19

Table 3: Task dimensions for HumanoidBench.

D More Experimental Results

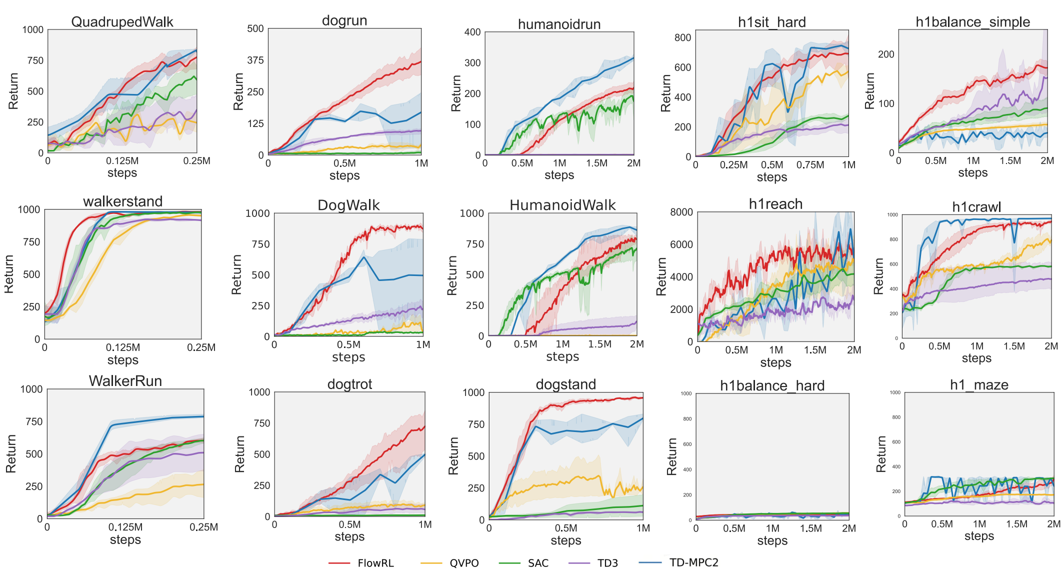


Figure 7: Experimental results are reported on 12 tasks drawn from HumanoidBench and DMC-hard, 3 tasks from DMC-easy & middle.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: Yes, the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, the paper discusses its limitations in the appendix. Although competitive performance is achieved without explicit exploration mechanisms, the exploration regularization mechanism remains an important direction for future work.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: Yes, the paper provides the full set of assumptions and complete proofs for all theorems in the appendix.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Yes, the paper fully discloses all information needed to reproduce the main experimental results. Pseudocode is provided in the main text, and all experimental settings, hyperparameters, and baseline details are included in the appendix.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Yes, the code will be made available after the open-source approval process is completed.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, all hyperparameters and experimental setup details necessary to understand the results are provided in the appendix.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Yes, for all model-free algorithms, five random seeds are used, and for model-based algorithm, three random seeds are used. All results are presented as mean ± standard deviation.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Yes, the appendix provides detailed information about the specific computational devices used for the experiments.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We make sure the code was anonymous

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Yes, the appendix discusses both potential positive societal impacts and limitations of the work.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Yes, all sources of data and code are properly credited in the appendix, with licenses and terms of use clearly indicated.

    Guidelines:

    - The answer NA means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.

- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer:[NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification:

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.

- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.