Horizon Adaptive Offline Policy Learning via Value Stitching

Kexin Zheng[1] **Xianyuan Zhan**[2†] **Xintao Yan**[1†]
[1] The University of Hong Kong [2] Tsinghua University
whiterr-zheng@connect.hku.hk
zhanxianyuan@air.tsinghua.edu.cn xintaoy@hku.hk

Abstract

Learning accurate value functions plays a decisive role for reinforcement learning (RL) agents to solve long-horizon, complex tasks. Conventional temporal-difference (TD) learning objectives suffer from value-estimation bias that accumulates over the horizon, while extended-horizon modeling methods, such as n-step TD backups and Q-chunking, adopt a rigid, fixed-horizon value-modeling recipe that is often not flexible enough to capture complex value structures in long-horizon, multi-stage tasks. In this paper, we show that enabling value updates with dynamic horizon composition can yield a strong offline policy learning scheme. Our method, *Horizon Adaptive Offline Policy Learning via VAlue STitching* (***VAST***), replaces fixed-horizon backups with recursive, horizon-adaptive value composition. Its key ingredient is to couple value optimization with a future state- and horizon-length-conditioned ***auxiliary value function*** that is learned through direct data supervision, and a ***stitching policy*** that optimally selects the reward-maximizing horizon length and future sub-goal to achieve horizon-adaptive value stitching. This design enables direct estimation and compositional "stitching" of variable-length returns grounded in actionable sub-goal states, providing an accurate and greedily exploitable value-supervision signal for offline policy optimization. Across 50 tasks on OGBench, *VAST* outperforms fixed-step, extended-horizon methods, and generative-value offline RL baselines, achieving strong performance particularly in high-complexity, long-horizon decision-making tasks. The official implementation is available at: https://github.com/Whiterrrrr/value-stitching.

1 Introduction

The quintessence of reinforcement learning (RL) lies in value [1–5]: it evaluates how present choices shape future consequences, provides the key signal for value judgment [6–8], and ultimately steers both the direction of optimization and the quality of the learned policy [9]. However, in long-horizon, complex task settings, conventional temporal-difference (TD) learning is highly susceptible to error compounding over the horizon due to myopic bootstrapping iterations on compressed scalar targets, which often yields biased value estimates and inefficient learning [6, 9–13]. To mitigate this limitation, recent works have started pursuing horizon-extended backups [14–18], distributional value modeling [19–21], or structured value representations that encode temporal relationships [22]. While extending the effective horizon can improve value scalability by supplying richer and more structured supervision signals [13], existing approaches, which are typically formulated as fixed-step TD backups [14] or chunked action/reward sequences [14, 15, 17, 18], still exhibit deficiencies due to their rigid, fixed-horizon parameterization.

†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

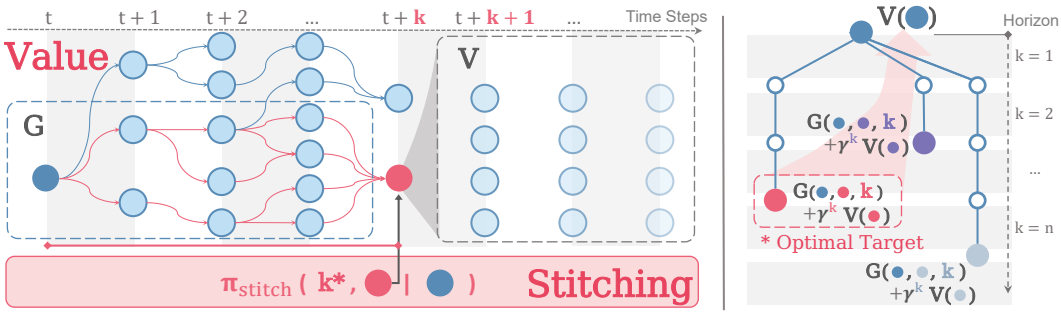

Figure 1: ***VAST*** **compositionally stitches cumulative returns between adaptive-length, actionable sub-goals, thereby enhancing value learning optimality and stability.** (*Left*): *VAST* leverages the *horizon-based cumulative return* (G-function) to measure the expected discounted return between two reachable states over a specific horizon, coupled with a stitching policy performing horizon-aware optimal sub-goal selection. (*Right*): *VAST* enables principled horizon adaptive value supervision guided by G-function with learnable backups.

In stark contrast, human decision-making in real-world scenarios exhibits a high degree of dynamism and adaptability regarding credit assignment over task horizons. When planning for distant objectives, humans typically only conceptualize coarse-grained sub-goals with a rough estimate of horizon length (e.g., walk straight for about 100m and then turn right); while for immediate and critical decisions, they will focus exclusively on proximal, actionable consequences (e.g., decelerate and move left to avoid an obstacle). This mixed-horizon modeling requirement is akin to the idea of the "option" framework [23, 24], which solves a semi-Markov decision process by optimizing temporally extended actions (i.e., macro-actions) with a terminal condition, thereby securing global optimality through localized decisions [25–27]. However, existing option-based methods remain structurally heavy and cumbersome to solve complex practical control problems. To simplify the modeling framework, many studies also resort to the hierarchical RL framework [13, 28–30], which jointly optimizes a high-level and a low-level policy at two distinct time scales. However, the hierarchical frameworks still can only capture two fixed modeling horizons, and the truly adaptive and practically deployable mixed-horizon offline RL paradigms remain largely underexplored.

To address this challenge, we propose ***VAST*** (illustrated in Figure 1), a horizon-adaptive offline RL framework that equips value and policy learning with temporal adaptivity, while simultaneously offering improved value learning stability and policy optimality. At its core, *VAST* integrates value optimization with a future-state- and horizon-conditioned ***auxiliary value function*** (referred to as *horizon-based cumulative return*, G-function) and a ***stitching policy*** (π_{stitch}) that adaptively selects optimal sub-goal states and corresponding stitching horizons. Specifically, the G-function estimates the expected discounted return between arbitrary reachable state pairs under a specified horizon, enabling direct evaluation of actionable sub-goals. This naturally gives rise to a new *stitching Bellman operator* that, together with the stitching policy, identifies the optimal target state for horizon-adaptive value backup. For policy extraction, *VAST* decomposes return maximization into the aforementioned ***stitching policy*** π_{stitch}, which selects horizon-aware sub-goals that best steer behavior toward high-return futures, and an ***execute policy*** π_{exec}, which is supervised trained directly on offline data and further optimized at test time via rejection sampling. The overall design enables both direct estimation and compositional stitching of returns anchored at actionable sub-goals, yielding supervision that is accurate, stable, and greedily exploitable for both value learning and low-level control. Across 50 OGBench [31] tasks under the offline RL setting, *VAST* performs the best or on par with n-step backup, Q-chunking methods, and generative-value-based baselines, with especially pronounced performance gains in complex long-horizon tasks.

2 Preliminaries

Problem formulation. We consider the RL problem in a Markov Decision Process (MDP) [6], defined by a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the operator that represents the transition dynamics, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ denotes the reward function, and $\gamma \in (0, 1)$ is the discount factor. The goal of the RL problem is to find a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes the expected return: $\mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k \cdot r(s_k, a_k)\right]$. In this work,

we focus on the offline RL setting, the goal of which is to learn a policy under the fixed dataset $\mathcal{D} = \{s_i, a_i, r_i, s_{i+1}\}_{i=1}^N$ collected by a behavioral policy μ.

Single-step and multi-step value backups. Approximate dynamic programming [32] is a central paradigm in RL, which maintains an action-value function $Q(s,a)$, or optionally a state-value function $V(s)$ for policy π under a given reward function $r : \mathcal{S} \to \mathbb{R}$, defined as $Q(s,a) := \sum_{t\geq 0} \gamma^t \cdot \mathbb{E}_{a_t\sim\pi}[r(s_t,a_t)|s,a,\pi]$ and $V(s) := \sum_{t\geq 0} \gamma^t \cdot \mathbb{E}_{a_t\sim\pi}[r(s_t,a_t)|s,\pi]$ respectively, with a further definition of the advantage function as $A(s,a) = Q(s,a) - V(s)$. In the offline RL setting, the value functions are typically learned with the following policy evaluation operator [33–36]:

$$(\mathcal{T}^\pi Q)(s,a) := r(s,a) + \gamma \cdot \mathbb{E}_{\mathcal{P}(s'|s,a)}\mathbb{E}_{a'\sim\pi}\left[Q(s',a')\right], \tag{1}$$

$$(\mathcal{T}^\pi V)(s) := \mathbb{E}_{a\sim\pi}\left[r(s,a) + \gamma \cdot \mathbb{E}_{\mathcal{P}(s'|s,a)}\left[V(s')\right]\right]. \tag{2}$$

The Q and V-function are typically learned by the conventional single-step objectives $\min_Q J(Q) = \frac{1}{2}\mathbb{E}_{(s,a)\sim\mathcal{D}}\left[(\mathcal{T}^\pi Q - Q)(s,a)^2\right]$ and $\min_V J(V) = \frac{1}{2}\mathbb{E}_{s\sim\mathcal{D}}\left[(\mathcal{T}^\pi V - V)(s)^2\right]$, respectively. To mitigate error accumulation and improve learning stability, some early and recent works enforce stronger supervision by adopting the n-step backup [6, 13, 14, 37–39]:

$$(\mathcal{T}^\pi_{\text{n-step}})Q(s_t,a_t) := \sum_{t'=t}^{t+n-1}\left[\gamma^{t'-t} r(s_{t'},a_{t'})\right] + \gamma^n \cdot \mathbb{E}_{s_{t+n}\sim\mathcal{P}}\mathbb{E}_{a_{t+n}\sim\pi}\left[Q(s_{t+n},a_{t+n})\right], \tag{3}$$

$$(\mathcal{T}^\pi_{\text{n-step}}V)(s_t) := \mathbb{E}_{a_t\sim\pi}\left[\sum_{t'=t}^{t+n-1}\left[\gamma^{t'-t} r(s_{t'},a_{t'})\right] + \gamma^n \cdot \mathbb{E}_{s_{t+n}\sim\mathcal{P}}\left[V(s_{t+n})\right]\right]. \tag{4}$$

Chunked value function. Another series of methods tackles the value bootstrapping scheme with action chunking [15–18], which extends the value backup from the action space to the action sequence level:

$$(\mathcal{T}^\pi_{\text{chunking}}Q)(s_t,a_{t:t+n}) := \sum_{t'=t}^{t+n-1}\left[\gamma^{t'-t} r(s_{t'},a_{t'})\right] + \gamma^n \cdot \mathbb{E}_{s_{t+n}\sim\mathcal{P}}\mathbb{E}_{a_{t+n:t+2n}\sim\pi}\left[Q(s_{t+n},a_{t+n:t+2n})\right]. \tag{5}$$

Compared with the n-step backup, the chunked value naturally suits the popular action-chunking design in imitation learning [40] and mitigates off-policy bias in previous methods [17]. However, both paradigms still suffer from some deficiencies due to the fixed horizon length modeling. n-step backups are inherently conservative in value estimation (the n-step return is estimated from the data collection policy rather than the optimal policy), whereas action chunking greatly expands the action space, creating potential optimization challenges.

Option-value function. The option framework [23, 24] models a Markovian option $\omega \in \Omega$ as a triple $(\mathcal{I}_\omega, \pi_\omega, \beta_\omega)$, where $\mathcal{I}_\omega \subseteq \mathcal{S}$ is the initiation set, $\pi_\omega : \mathcal{S}\times\mathcal{A} \to [0,1]$ is the intra-option policy over primitive actions, and $\beta_\omega : \mathcal{S} \to [0,1]$ is the termination function. Augmenting an MDP with a fixed set of options Ω (by a policy over options $\pi_\Omega(\omega \mid s)$) induces a semi-Markov decision process (SMDP) [23, 41], on which the option-value function is defined as $Q_\Omega(s_t,\omega) := \mathbb{E}_{\pi_\omega}\left[\sum_{t'=t}^{t+k-1}\gamma^{t'-t} r(s_{t'},a_{t'}) + \gamma^k V_\Omega(s_{t+k}) \mid s_t,\omega\right]$, where we use k to denote the random duration at which ω terminates according to β_ω, and V_Ω is the value of options at the call-and-return state s_{t+k}; the expectation is taken jointly over the action sequence and the termination event. While this formulation yields a principled treatment of temporally extended behavior, existing option-based approaches remain architecturally heavy, which poses considerable challenges for practical control tasks [25–27, 42]. In this work, we borrow the insight from the option framework, but establish a more effective and simpler value learning framework with flexible horizon length modeling capability. More discussions of various value learning schemes and hierarchical RL frameworks are provided in Section 6.

3 Modeling Cumulative Return for Adaptive Horizon Length

The Bellman updates can accumulate substantial error through recursive bootstrapping on intermediate predicted targets (Eq. (1)), especially in long-horizon tasks [6, 10, 12, 13, 17, 18]. As discussed

in the previous section, multi-step value learning schemes such as n-step backups (Eq. (3)) and chunked value function (Eq. (5)) can strengthen supervision by extending the horizon of grounded reward signals, but are limited by fixed-step backups. In the following example, we illustrate the pitfalls of fixed-step value updates and then present a neat solution for horizon-adaptive value stitching.

The limitation of fixed-horizon modeling. Central to the identified problem of fixed-horizon value-learning is that, while it may yield more robust value estimates, it is poorly suited to extracting policies for real-time, stepwise execution. When the backup sequence length is mismatched, crucial single-step signals can be washed out by sequence-level averaging, leading to the loss of precisely the information needed for accurate immediate decisions. This intuitive failure mode is readily illustrated by a simple toy example: the 4×4 grid navigation task in Figure 2. The agent starts at the bottom-left corner and aims to reach the goal at the top-right corner by moving only upward or rightward. Rewards for each reach are encoded by cell color shown in the data subgraph: dark red denotes high reward (e.g., 2), light red denotes lower reward (e.g., 1), green skull denotes penalty (e.g., -2), and blank cells have zero reward. The dataset contains only suboptimal trajectories (the green and purple paths).

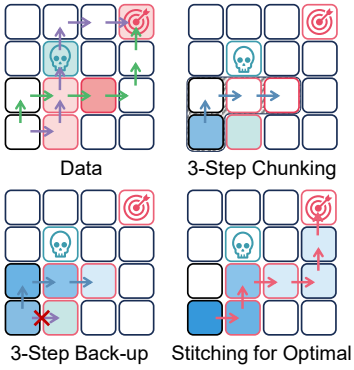

Figure 2: **4x4 grid navigation.** Toy example requiring navigation from the bottom-left to the top-right. The data include sub-optimal paths only (**green path** and **purple path**); the environment is specified by **dark red**, **light red**, and **skull** blocks, indicating high-, low-, and punitive-reward. The **depth of blue in blocks** indicates the value estimates from fixed-step/chunked value backups, and the optimal stitching value.

Under an improper fixed horizon (e.g., the 3-step horizon in this example), the agent fails to recover optimal behavior. The intensity of the blue shading indicates the learned value under each method, with darker regions corresponding to higher values. Chunked value updates favor the first three transitions along the green path, but fail to guide the agent to the destination. In contrast, n-step backups suppress the transition from $(0, 0)$ to $(1, 0)$ (the first purple transition in data) because it is contaminated by the later negative reward at $(1, 2)$; and notably, both fixed-step methods lead to a negative value on $(1, 0)$ with a discount factor $\gamma \approx 1$, shown with a skull shading. However, the optimal solution requires stitching together favorable short segments from different trajectories (red path). Refer to Appendix C for further details. Although deliberately simple, this toy example can be viewed as a local subproblem embedded within a much larger maze. It shows that while coupling value updates to a fixed horizon can better stabilize long-horizon learning scenarios, it simultaneously obscures critical short-range decisions, thereby trapping learning in poor local optima. Stability should not be bought at the cost of erasing decisive short-term signals, motivating adaptive horizon modeling that preserves both robustness and flexibility.

Towards adaptive horizon composition. The failure of fixed-horizon methods raises a fundamental question: *how should the backup horizon be selected optimally?* Addressing this issue requires moving beyond value estimates rigidly tied to transition sequences in the data buffer. Instead, we need a more general operator that can evaluate the cumulative return between any pair of states under a *dynamically chosen horizon*. This insight motivates the introduction of the following definition:

Definition 1 (Horizon-Conditioned Cumulative Return)**.** *Let $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \dots)$ be the trajectory induced by a policy π starting at state $s_0 = s$. For a target state s^+ reached exactly at step k (i.e., $s_k = s^+$), we define the horizon-conditioned cumulative return as:*

$$G(s, s^+, k) := \mathbb{E}_{\tau \sim \pi}\left[\sum_{t=0}^{k-1} \gamma^t r(s_t, a_t) \;\middle|\; s_0 = s,\ s_k = s^+\right]. \tag{6}$$

Given $|r(s, a)| \le R_{\max}$ and $\gamma \in (0, 1)$, then G is strictly bounded as $|G(s, s^+, k)| < \frac{R_{\max}}{1-\gamma}$.

We refer to $G(s, s^+, k)$ as the *Horizon-Conditioned Cumulative Return Function*. By conditioning on both the terminal state s^+ and the exact horizon k, G defines a value mapping relationship between arbitrary k-step reachable state pairs and the policy-induced distribution. This formulation

breaks the dependence of value targets on fixed local backup segments and further supports flexible temporal composition across extended sub-trajectory segments, as shown in Theorem 1, thereby forming the foundation for adaptive horizon selection in value learning.

Theorem 1 (Temporal Compositionality)**.** *For any target state s_k reachable from s at step k, and any intermediate state s_i visited at step i with $0 < i < k$ along a trajectory induced by policy π, the horizon-conditioned cumulative return admits the decomposition:*

$$G(s, s_k, k) = \mathbb{E}_{i\sim\text{Unif}\{1,\dots,k-1\},\, s_i\sim\pi}\left[G(s, s_i, i) + \gamma^i G(s_i, s_k, k-i)\right]. \tag{7}$$

The proof is provided in Appendix B.1.

Theorem 1 formalizes the key principle underlying temporal value composition. It shows that a long-horizon return can be decomposed into a sequence of shorter and composable return segments. This property is particularly important for a constructive function that learns value relationships across multiple temporal scales among reachable states.

Learning objective of G-function. To learn a parameterized G-function $G_\psi(s, s^+, k)$ from an offline dataset $\mathcal{D}$, we optimize two complementary objectives: (i) a Monte-Carlo regression objective for direct supervision, and (ii) a temporal compositional objective induced by Theorem 1 to enforce multi-scale consistency:

$$\mathcal{L}_{\text{MC}}(\psi) = \mathbb{E}_{\tau\sim\mathcal{D},\, k\sim\text{Unif}\{1,\dots,K\}}\left[\left(G_\psi(s, s_k, k) - \sum_{t=0}^{k-1}\gamma^t r(s_t, a_t)\right)^2\right], \tag{8}$$

$$\begin{aligned}\mathcal{L}_{\text{comp}}(\psi) =& \mathbb{E}_{\tau\sim\mathcal{D},\, k\sim\text{Unif}\{2,\dots,K\},\, i\sim\text{Unif}\{1,\dots,k-1\}}\Big[\Big(G_\psi(s, s_k, k) \\ &- \left(G_\psi(s, s_i, i) + \gamma^i G_\psi(s_i, s_k, k-i)\right)\Big)^2\Big],\end{aligned} \tag{9}$$

where trajectory τ is sampled uniformly from offline dataset $\mathcal{D}$, k is sampled uniformly from $\{1, \dots, K\}$ with K denoting the maximum horizon, starting state $s_0 = s$ is randomly sampled from the τ, and i is sampled uniformly from $\{1, \dots, k-1\}$. The overall objective is

$$\mathcal{L}_G(\psi) = \mathcal{L}_{\text{MC}}(\psi) + \lambda\mathcal{L}_{\text{comp}}(\psi). \tag{10}$$

The MC objective provides direct return supervision, while the compositional objective regularizes G_ψ to satisfy temporally consistent value composition, improving generalization across horizons. Once learned, G_ψ is no longer restricted to fixed backup segments from the offline data buffer, thereby enabling adaptive, target-state-anchored value maximization..

4 Horizon Adaptive Offline Policy Learning

With the G-function established in Section 3, we now turn to its use in offline policy optimization. To this end, we propose ***VAST***, which is built upon a novel horizon-adaptive stitching Bellman operator, as well as a hierarchical policy optimization scheme involving a *stitching policy* π_{stitch} and an *execution policy* π_{exec}. In the following, we describe the details of our algorithm design.

Stitching Bellman operator. Note that with the previously defined G-function, for a given policy π, we can equivalently express the state-value function V^π as (assuming sufficiently large K):

$$V^\pi(s) = \mathbb{E}_{k\sim\text{Unif}\{1,\dots,K\}, s_k\sim\Gamma^\pi(s,k)}\left[G(s, s_k, k) + \gamma^k \cdot V^\pi(s_k)\right]. \tag{11}$$

where $\Gamma^\pi(s, k)$ denotes the state distribution induced by rolling out with π starting from s for k steps. And in particular, when $k = 1$, the above reduces to the special one-step horizon case,

$$V^\pi(s_t) = \mathbb{E}_{s_{t+1}\sim\mathcal{P}(\cdot|s_t,a_t), a_t\sim\pi}\left[G(s_t, s_{t+1}, 1) + \gamma \cdot V^\pi(s_{t+1})\right], \tag{12}$$

which recovers the standard Bellman equation with one-step anchoring (as $G^\pi(s_t, s_{t+1}, 1) = r(s_t, a_t)$). This observation naturally motivates us to introduce a new *stitching* Bellman operator:

$$(\mathcal{T}^\pi_{\text{stitching}} V)(s) = \max_{k\in\{1,\dots,K\}, s_k\sim\Gamma^\pi(s,k)} G(s, s_k, k) + \gamma^k \cdot V(s_k). \tag{13}$$

Specifically, the $\max$ operator is a *joint* maximization over the horizon $k \in \{1, \ldots, K\}$ and the target states s_k in the support of the k-step reachable distribution $\Gamma^\pi(s, k)$. Therefore, only the feasible set contains state-horizon pairs (k, s_k) that are actually reachable from s in exactly k steps under π will be evaluated. Repeatedly applying this operator on V will converge to a unique fixed point V^*, as it can be proved to be a γ-contraction mapping:

Theorem 2 (Contraction Mapping of the Stitching Bellman Operator)**.** *For any two bounded value functions V_1, V_2, there exists a horizon length $\tilde{k} \sim \{1, \cdots, K\}$ such that the operator $\mathcal{T}_{stitch}$ is a γ-contraction in the infinity norm:*

$$\|\mathcal{T}_{stitch} V_1 - \mathcal{T}_{stitch} V_2\|_\infty \leq \gamma^{\tilde{k}} \|V_1 - V_2\|_\infty \leq \gamma \|V_1 - V_2\|_\infty. \tag{14}$$

See Appendix B.2 for the proof. With the state-value function V estimated using the stitching Bellman operator, the action-value function Q can be learned via the standard Bellman backup:

$$(\mathcal{T}^\pi Q)(s, a) := r(s, a) + \gamma \cdot \mathbb{E}_{s' \sim P(\cdot|s)} \left[V(s') \right], \tag{15}$$

Value stitching via π_{stitch}. The stitching Bellman operator offers a new way to bootstrap the value estimates based on the optimal horizon lengths and target states. However, practically achieving such horizon-adaptive value learning in the offline setting requires some additional designs. In this study, we introduce a stitching policy $\pi_{\text{stitch}}(\cdot|s) : \mathcal{S} \to (\mathbb{Z}^+_{\leq K}, \mathcal{S})$ that simultaneously selects 1) the best horizon k for value stitching and 2) the reward-maximizing sub-goal state s_k based on current state s. Based on Eq.(13), the offline learning objective for the V-function can be written as:

$$V = \arg\min_V \mathbb{E}_{s \sim \mathcal{D}} \Big[\Big[\max_{(k, s_k) \sim \pi_{\text{stitch}}(\cdot|s)} \underbrace{G(s, s_k, k) + \gamma^k V(s_k)}_{V^\pi_{\text{target}}} \Big] - V(s) \Big]^2 \tag{16}$$

The above value objective adaptively regresses the value target based on the optimal horizon length specified by the G-function, thereby achieving horizon-adaptive *value stitching*. However, evaluating the $\max$ operator can be difficult for the continuous state space. Therefore, we leverage the expectile regression objective as introduced in IQL [35] to mitigate this difficulty:

$$\mathcal{L}_V = \mathbb{E}_{s \sim \mathcal{D}, (k, s_k) \sim \pi_{\text{stitch}}(\cdot|s)} \left[L_2^\tau \left(V^\pi_{\text{target}} - V(s) \right) \right], \tag{17}$$

where $L_2^\tau(u) = |\tau - \mathbb{I}(u < 0)|u^2$, with $\tau > 0.5$ serving as the expectile parameter. V^π_{target} is evaluated at $(k, s_k) \sim \pi_{\text{stitch}}(\cdot \mid s)$. The policy weights below use $V^{\mathcal{D}}_{\text{target}}$, the same expression $G(s, s_k, k) + \gamma^k V(s_k)$ on a same-trajectory pair $(k, s_k) \sim \mathcal{D}$.

With the value learning objective established, we also need a simple approach to optimize the stitching policy π_{stitch} while adhering to the data distribution. Specifically, we choose the advantage weighted regression (AWR) [43] to extract π_{stitch}, which is widely adopted in many offline RL methods [35, 36, 44–46]. It can be perceived as regressing the optimal solution of an RL objective that maximizes the advantage value with a reverse Kullback-Leibler (KL) divergence constraint on the data distribution [36, 43]. But in our case, we evaluate and maximize over a special *state-advantage* conditioned on both horizon length and sub-goal state, i.e.,

$$\mathcal{L}_{\pi_{\text{stitch}}} = \mathbb{E}_{(s, k, s_k) \sim \mathcal{D}} \left[\exp\left(\beta \cdot \left(V^{\mathcal{D}}_{\text{target}} - V(s) \right) \right) \cdot \log \pi_{\text{stitch}}(k, s_k | s) \right], \tag{18}$$

Moreover, to further enhance the expressiveness and generalization capability of the stitching policy, as well as to improve the robustness of value backups, we parameterize π_{stitch} as a diffusion/flow model [47–49]. Concretely, π_{stitch} jointly predicts the optimal horizon k and sub-goal s_k in the continuous space $(\mathbb{R}_+, \mathcal{S})$. Let ϵ_{stitch} denote the neural network parameterizing the stitching policy, which is trained to predict the injected noise over the dataset samples. It has been shown and adopted in many diffusion-based RL studies [45, 46, 50, 51] that the following weighted diffusion loss can be used to optimize the above AWR-style objective:

$$\mathcal{L}_{\epsilon_{\text{stitch}}} = \mathbb{E}_{t \sim U[0,1], \epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I}), (s, k, s_k) \sim \mathcal{D}} \left[\exp\left(\beta \cdot \left(V^{\mathcal{D}}_{\text{target}} - V(s) \right) \right) \cdot \left\| \epsilon - \epsilon_{\text{stitch}} \left(k^t, s_k^t, s, t \right) \right\|^2 \right] \tag{19}$$

where $(k^t, s_k^t) = \alpha^t (k, s_k) + \sigma^t \epsilon$ with α^t and σ^t being predefined noise schedules. Empowered by the stitching policy, *VAST* not only enables effective state-value learning directly from offline data with only state-dependent transitions, but also flexibly extracts globally relevant goal information for better potential downstream utilization.

Optimized action extraction via π_{exec}. With the optimal sub-goal state predicted from the learned stitching policy π_{stitch}, we can learn a lower-level *execution policy* $\pi_{\text{exec}} : \mathcal{S} \times \mathcal{S} \to \mathcal{A}$ for control under its guidance. Due to the unique learning procedure of π_{stitch}, it can provide both reachable and reward-maximizing sub-goals. This is akin to the human decision-making process that performs both coarse global-level planning as well as adaptive short-horizon adjustment. We learn $\pi_{\text{exec}}(a|s, s^+)$ as a flow policy in a simple goal-conditioned imitation learning manner that directly supervises it with offline data samples, conditioned on both current state s and randomly sampled future goal state s^+ in the same data trajectory. This bypasses the out-of-distribution (OOD) issue during training while enjoying a stable learning process. Specifically, we omit k to decouple π_{exec} from the accuracy of π_{stitch}, thereby avoiding a train-inference mismatch, where at training time k comes from sub-optimal data, while at inference time it comes from the optimized π_{stitch}.

To further improve the optimality of the output action, similar to previous methods [52, 53], we filter the sampled actions with test-time rejection sampling conditioned on the sub-goal $s_k \sim \pi_{\text{stitch}}(\cdot|s)$:

$$a^\star \triangleq \arg\max_{a \in \{a^{(1)}, \ldots, a^{(N_{\text{rej}})} \sim \pi_{\text{exec}}(\cdot|s, s_k)\}} Q(s, a), \tag{20}$$

where N_{rej} is the action number to be sampled. Q-function is learned by minimizing the standard Bellman error with Eq.(15). The above formulation performs implicit Q-value maximization by re-sampling the distribution learned by the expressive goal-conditioned BC flow policy, with instructive and advantageous sub-goal information additionally injected. See Appendix G for more discussions, and Appendix E for additional implementation details.

5 Experiments

Briefly, *VAST* employs *an auxiliary value function* to improve *(1) value-learning* and further the *(2) policy-execution optimality and robustness*. Accordingly, we expect *VAST* to achieve strong performance on long-horizon complex planning tasks, outperforming strong value-scaling approaches.

Environmental setup. We evaluate *VAST* on the OGBench task suite [31]. OGBench is designed for goal-conditioned offline RL, with challenging robotic locomotion and manipulation tasks, including whole-body humanoid control, maze navigation, and object manipulation. Our experiments are conducted in the `singletask` variant for offline RL, covering 30 state-based tasks across 6 widely used domains, denoted as *Basic*, and 20 state-based tasks across 4 highly challenging domains that increase the planning complexity, denoted as *Hard*, with **a total of 50 tasks on OGBench**. See Appendix D.1 for environmental details.

Baselines. We select 8 baselines for comprehensive evaluation, which could be categorized into value learning methods with 1-step/multi-step TD, chunking, and generative value:

- *1-step TD.* We select 1) *IQL* [35]: a typical weighted regression offline RL method with Gaussian policy and expectile regression for value learning; 2) *FQL* [52]: a behavior-regularized actor-critic (AC) variant that uses flow policy distillation and shows strong performance on OGBench; 3) *IFQL* [52]: a variant of *IQL* that uses flow policy with rejection sampling during inference.
- *Multi-step TD.* We select 1) *FQL-n* and 2) *IFQL-n*, the variants of *FQL* and *IFQL* with fixed-n-step value backups. Following Li et al. [17], we fix the horizon to 5.
- *Chunking.* We select the most typical chunked value baseline *QC* [17]: Q-learning on a fixed-size action sequence space.
- *Generative value.* To illustrate *VAST*'s effectiveness on value learning and guidance, we further select the baselines that specifically focus on value scaling. This includes 1) *VALUE-FLOWS* [21]: a novel distributional RL method that estimates return variance using a flow-matching objective and learns a flow policy; 2) *FLOQ* [20]: it parameterizes the critic as a learned velocity field, estimates the Q-value through numerical integration, and learns a flow policy.

See Appendix D.3 for additional baseline information.

Experimental results and ablations. We evaluate each method with a fixed number of offline updates (1M) and report the success rate. We average the scores across 8 random seeds on main

Table 1: **OGBench Results.** We report the aggregate score on all single tasks for each category, averaging over 8 random seeds. *VAST* achieves the best or near-best performance against other baselines across 10 domains, 50 tasks in total. The IQL results are reported from Park et al. [52], thereby eliminating the $\pm$ sign.

	1-step TD			n-step TD		Chunking	Generative		Stitching
(5 tasks each)	IQL	FQL	IFQL	FQL-n	IFQL-n	QC	VALUE-FLOWS	FLOQ	VAST
`scene`	28	57$\pm$7	46$\pm$12	42$\pm$8	32$\pm$2	60$\pm$1	60$\pm$5	57$\pm$5	61$\pm$3
`cube-single`	83	97$\pm$4	87$\pm$5	97$\pm$3	82$\pm$6	93$\pm$5	95$\pm$4	96$\pm$5	98$\pm$2
`cube-double`	7	30$\pm$8	12$\pm$5	4$\pm$3	4$\pm$3	67$\pm$7	61$\pm$11	49$\pm$16	75$\pm$6
`puzzle-4x4`	7	15$\pm$5	27$\pm$8	27$\pm$6	12$\pm$4	29$\pm$10	21$\pm$8	31$\pm$9	46$\pm$10
`antmaze-large`	53	82$\pm$7	32$\pm$17	80$\pm$9	27$\pm$7	6$\pm$5	48$\pm$15	93$\pm$6	81$\pm$7
`antmaze-giant`	4	11$\pm$16	0$\pm$2	37$\pm$16	0$\pm$0	0$\pm$0	3$\pm$3	29$\pm$24	20$\pm$6
Overall (*Basic*)	30	49$\pm$9	34$\pm$10	48$\pm$9	26$\pm$4	43$\pm$6	48$\pm$9	59$\pm$13	64$\pm$6
`cube-triple`	-	1$\pm$4	1$\pm$1	0$\pm$0	0$\pm$0	4$\pm$5	13$\pm$10	4$\pm$5	17$\pm$8
`cube-quadruple`	-	0$\pm$0	0$\pm$0	0$\pm$0	0$\pm$0	0$\pm$0	0$\pm$0	0$\pm$0	1$\pm$1
`puzzle-4x5`	-	15$\pm$3	16$\pm$7	18$\pm$1	14$\pm$4	20$\pm$0	12$\pm$4	13$\pm$3	20$\pm$0
`puzzle-4x6`	-	4$\pm$5	0$\pm$0	4$\pm$9	3$\pm$5	4$\pm$4	3$\pm$6	3$\pm$3	8$\pm$17
Overall (*Hard*)	-	5$\pm$4	4$\pm$4	6$\pm$5	4$\pm$3	7$\pm$3	7$\pm$6	5$\pm$3	12$\pm$9

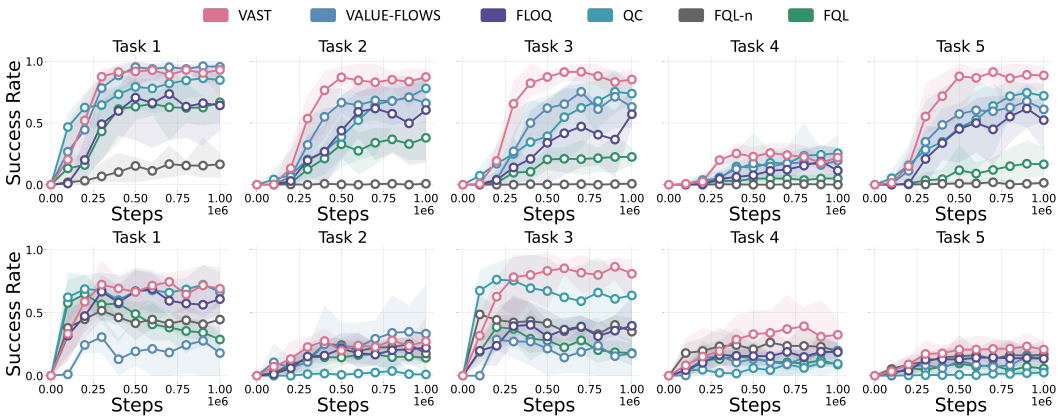

Figure 3: **Learning curves.** (*Top*): `cube-double-play`. (*Bottom*): `puzzle-4x4-play`.

results, with $\pm$ indicating standard deviations in the tables. For each ablation, we average the results on 3 random seeds. We analyze the experiments with the following questions:

How does VAST perform? Table 1 summarizes the aggregate results on OGbench. *VAST* achieves the best or near-best performance in both domain-wise averages and the overall score, with particularly pronounced gains on manipulation domains such as `scene`, `cube`, and `puzzle`. Notably, *VAST* is the best or tied-best method on every *Hard* domain, achieving an overall score that is 1.7$\times$ that of the best baseline. This demonstrates its ability to effectively exploit dataset transitions and scale to long-horizon decision-making problems. Figure 3 presents the offline learning curves. *VAST* consistently outperforms competing methods on most tasks, and its sustained upward trend in the *Hard* setting further highlights its scalability. To better understand its behavior, we visualize the ground-truth trajectory together with the future states predicted by *VAST* on `cube-triple-play-task3` in Figure 4. The predicted horizon follows a plausible sequence of phased atomic intentions, suggesting that the policy selects an effective intermediate target $k, s_k \sim \pi_{\text{stitch}}$ and thereby progresses toward stable, value-maximizing actions. Full per-task results and learning curves are provided in Appendix F.

Does VAST benefit from wider networks and larger batches? Figure 5 reports the performance of *VAST* on `cube-triple-play` with layer widths and batch sizes of 512 and 1024. Performance generally improves with larger layer widths and batch sizes, and setting both to 1024 yields the best results on most tasks, indicating strong potential for scaling with model capacity and compute. However, this improvement comes at a higher computational cost: increasing both nearly triples the

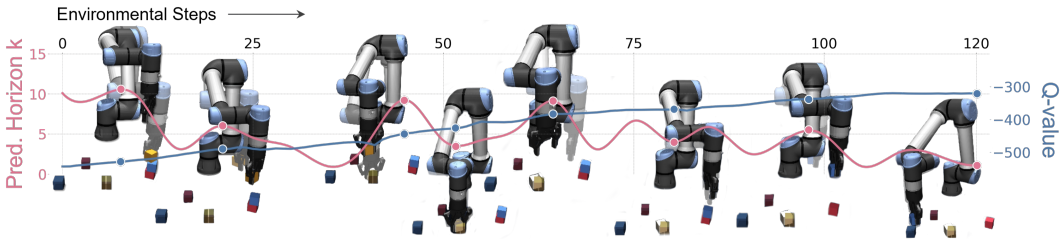

Figure 4: **Visualization on `cube-triple-play-task3` (pop_from_stuck).** The solid robot arm shows the current pose, while the transparent arm shows the sub-goal.

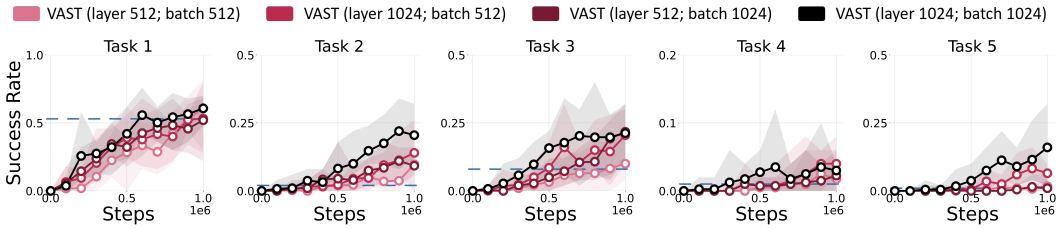

Figure 5: **Performance on `cube-triple-play` with different layer widths and batch sizes.** The blue dashed line marks the best baseline on each task. *VAST* benefits from larger layer widths and batch sizes.

GPU time for 1M training steps (see Appendix H.1). For a practical trade-off between performance and efficiency, we report the main results with the default layer width and batch size of 512.

Does the horizon-length limit affect VAST? Figure 6 (*left 1-2*) compares the performance of *VAST* under different horizon limits K in value learning (Eq. (8-9), Eq. (11)). Overall, performance improves as K increases from 5, indicating that longer horizons provide greater flexibility for selecting informative targets during learning. Intuitively, a larger K expands the candidate set of future targets and therefore increases the chance of identifying a high-value intermediate target (k, s_k). In most manipulation domains, *VAST* achieves its best performance around $[10, 50]$, strongly relying on the task setting. Further increasing K leads to a mild performance drop, suggesting that excessively long horizons may make target estimation more challenging and place higher demands on the expressivity of the G-function.

How important is the compositional objective? Figure 6 (*left 3-4*) reports the performance of *VAST* under different compositional weights λ (Eq. 10)). Performance slightly improves as λ increases within an appropriate scale, demonstrating its benefit. While the MC objective (Eq. (8)) provides direct supervision from dataset transitions, the compositional objective further encourages temporal consistency of the learned G-function and improves its generalization beyond observed trajectories. As a result, it supplies more reliable value estimates for downstream policy optimization. When λ becomes too large, performance decreases, suggesting that the compositional term should be balanced with the MC objective; overly strong self-supervised regularization may dominate the learning signal and lead to less informative representations, such as degenerate fixed-point solutions.

Is two-level reward maximization necessary? We compare different rejection sampling strengths for π_{exec} in Figure 6 (*left 5*). When $N_{\text{rej}} = 1$, the framework effectively relies on π_{stitch} alone, where reward maximization is mainly achieved through goal selection and the goal-conditioned BC policy directly executes the selected goal. Increasing N_{rej} improves performance, indicating that additional

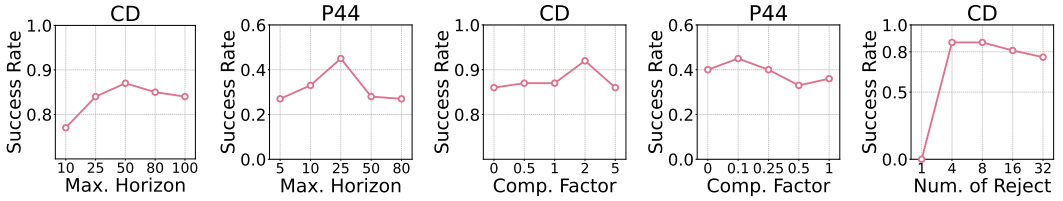

Figure 6: **Ablations on important hyperparameter**. We provide the horizon length ablation (*left 1-2*), the compositional factor ablation (*left 3-4*), and the rejection sampling strength ablation (*left 5*) using default tasks on domain `cube-double-play` and `puzzle-4x4-play`.

reward-aware selection at the execution level is a useful complement coupled with high-level goal selection. These results validate the two-level reward-maximization design of *VAST*.

6 Related Works

Offline RL. Offline RL aims to learn an optimal policy from the given dataset without environmental interaction. A central challenge is distributional shift, where value functions may assign overly optimistic estimates to out-of-distribution (OOD) actions, leading to poor policy improvement [54]. Early methods mainly address this issue from two perspectives. The first line constrains the learned policy to stay close to the behavior policy [55–60]. The second line directly regularizes the value function by penalizing overestimated OOD queries [34, 61–64]. More recent studies suggest that, beyond conservative value estimation, effective policy extraction is often the key factor that determines final performance [9]. This insight has motivated lightweight in-sample learning methods [28, 35, 36, 65], which are stable, simple, and competitive in absolute performance. In parallel, generative models such as diffusion [66–68] and flow-based models [69] have been introduced to improve policy expressivity and generalization [45, 46, 50, 53, 70–76]. Together, these developments have made offline RL increasingly practical for real-world applications [77, 78].

Value learning in RL. Value learning is central to RL, as it determines the performance ceiling of the learned policy; meanwhile, it remains a key bottleneck for scaling RL to complex and long-horizon problems [13]. Existing efforts to improve value learning can be broadly grouped into three directions: (i) stabilizing and strengthening value estimation through multi-step reward propagation, which includes multi-step TD backups in the action space [13, 14], as well as methods that couple value estimates over fixed-length action or trajectory chunks [15–18]; (ii) distributional approaches model the full return distribution rather than only its expectation, either explicitly learn a discrete return distribution with projection operators [19, 79, 80], or implicitly represent return distributions using expressive generative models [20, 21], including diffusion [66–68] and flow matching [49, 69] models; (iii) scaling the parameterization of value functions with expressive architectures, such as Transformers and Mixtures of Experts (MoEs) [81–83].

Hierarchical and option in RL. The option framework was introduced to improve the scalability of RL by abstracting temporal dynamics into reusable high-level behaviors [23–27]. In this framework, an option is typically defined by an initiation condition, a termination condition, and an intra-option policy. This formulation enables agents to reason and act over extended time scales, and has naturally led to a broad line of hierarchical RL (HRL) methods [13, 22, 25–29, 42, 84–88]. Despite this progress, HRL still faces a key optimization challenge lies in non-stationary learning, where the high-level policy must optimize over goals or options whose effects keep changing as the low-level policy is updated [30].

7 Conclusion and Future

In this work, we introduced *VAST*, a novel RL method that decouples value learning from fixed-step backups via an auxiliary value function that enables "stitching" operation. *VAST* defines a stitching Bellman operator and employs two-level policies to maximize reward. Its excellent performance on OGBench suggests a promising direction for scalable long-horizon decision-making with horizon-aware future identification. Looking forward, the core of *VAST* could be naturally combined with several diffusion/flow policy optimization methods, model-based methods, and representation learning methods, offering an exciting avenue for further improvement of horizon-adaptive RL. See Appendix H for more discussion on limitations and future work.

Acknowledgment

This work was supported by the Wuxi Research Institute of Applied Technologies at Tsinghua University under the Grant 20242001120, the Xiongan AI Institute, the Early Career Scheme (ECS) of the Research Grants Council of Hong Kong under Project No. HKU27206826, the University of Hong Kong Faculty Interdisciplinary Fund, and the University of Hong Kong Urban Systems Institute (HKU-USI) Fellowship Grant. We would also like to thank the NeurIPS review committee for their helpful comments and suggestions on this work.

References

[1] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988.

[2] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3):279–292, 1992.

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[4] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529 (7587):484–489, 2016.

[5] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[6] Richard S Sutton, Andrew G Barto, et al. Reinforcement learning: An introduction. 1(1), 1998.

[7] Eduardo Pignatelli, Johan Ferret, Matthieu Geist, Thomas Mesnard, Hado van Hasselt, and Laura Toni. A survey of temporal credit assignment in deep reinforcement learning. *Transactions on Machine Learning Research*, 2024.

[8] Chia-Chun Hung, Timothy Lillicrap, Josh Abramson, Yan Wu, Mehdi Mirza, Federico Carnevale, Arun Ahuja, and Greg Wayne. Optimizing agent behavior over long time scales by transporting value. *Nature communications*, 10(1):5223, 2019.

[9] Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[10] Tommi Jaakkola, Michael Jordan, and Satinder Singh. Convergence of stochastic iterative dynamic programming algorithms. *Advances in neural information processing systems*, 6, 1993.

[11] John Tsitsiklis and Benjamin Van Roy. Analysis of temporal-diffference learning with function approximation. *Advances in neural information processing systems*, 9, 1996.

[12] Kristopher De Asis, J Hernandez-Garcia, G Holland, and Richard Sutton. Multi-step reinforcement learning: A unifying algorithm. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[13] Seohong Park, Kevin Frans, Deepinder Mann, Benjamin Eysenbach, Aviral Kumar, and Sergey Levine. Horizon reduction makes rl scalable. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[14] George Konidaris, Scott Niekum, and Philip S Thomas. Td_gamma: Re-evaluating complex backups in temporal difference learning. *Advances in Neural Information Processing Systems*, 24, 2011.

[15] Younggyo Seo and Pieter Abbeel. Coarse-to-fine q-network with action sequence for data-efficient reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[16] Dong Tian, Onur Celik, and Gerhard Neumann. Chunking the critic: A transformer-based soft actor-critic with n-step returns. *arXiv preprint arXiv:2503.03660*, 2025.

[17] Qiyang Li, Zhiyuan Zhou, and Sergey Levine. Reinforcement learning with action chunking. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[18] Qiyang Li, Seohong Park, and Sergey Levine. Decoupled q-chunking, 2025.

[19] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International conference on machine learning*, pages 449–458. Pmlr, 2017.

[20] Bhavya Kumar Agrawalla, Michal Nauman, Khush Agrawal, and Aviral Kumar. floq: Training critics via flow-matching for scaling compute in value-based RL. In *NeurIPS 2025 Workshop: Second Workshop on Aligning Reinforcement Learning Experimentalists and Theorists*, 2025.

[21] Perry Dong, Chongyi Zheng, Chelsea Finn, Dorsa Sadigh, and Benjamin Eysenbach. Value flows. In *International Conference on Learning Representations (ICLR)*, 2026.

[22] Marlos C Machado, Andre Barreto, Doina Precup, and Michael Bowling. Temporal abstraction in reinforcement learning with the successor representation. *Journal of machine learning research*, 24(80):1–69, 2023.

[23] Richard S. Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1): 181–211, 1999. ISSN 0004-3702.

[24] Doina Precup. *Temporal abstraction in reinforcement learning*. University of Massachusetts Amherst, 2000.

[25] Martin Stolle and Doina Precup. Learning options in reinforcement learning. In *International Symposium on abstraction, reformulation, and approximation*, pages 212–223, 2002.

[26] Pierre-Luc Bacon, Jean Harb, and Doina Precup. The option-critic architecture. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2017.

[27] Matthew Riemer, Miao Liu, and Gerald Tesauro. Learning abstract options. In *The Thirty-second Annual Conference on Neural Information Processing Systems*, 2018.

[28] Haoran Xu, Li Jiang, Jianxiong Li, and Xianyuan Zhan. A policy-guided imitation approach for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

[29] Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. Hiql: Offline goal-conditioned rl with latent states as actions, 2023.

[30] Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

[31] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations (ICLR)*, 2025.

[32] Warren B Powell. *Approximate Dynamic Programming: Solving the curses of dimensionality*, volume 703. John Wiley & Sons, 2007.

[33] Ofir Nachum, Mohammad Norouzi, Kelvin Xu, and Dale Schuurmans. Bridging the gap between value and policy based reinforcement learning. In *The Thirty-first Annual Conference on Neural Information Processing Systems*, 2017.

[34] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

[35] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2021.

[36] Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization. In *International Conference on Learning Representations*, 2023.

[37] Christopher John Cornish Hellaby Watkins et al. Learning from delayed rewards, 1989.

[38] Jing Peng and Ronald J Williams. Incremental multi-step q-learning. In *Machine Learning Proceedings 1994*, pages 226–232. Elsevier, 1994.

[39] Philip S Thomas, Scott Niekum, Georgios Theocharous, and George Konidaris. Policy evaluation using the ω-return. *Advances in Neural Information Processing Systems*, 28, 2015.

[40] Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems*, 2023.

[41] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

[42] Alexander Sasha Vezhnevets, Simon Osindero, Tom Schaul, Nicolas Heess, Max Jaderberg, David Silver, and Koray Kavukcuoglu. Feudal networks for hierarchical reinforcement learning. In *International conference on machine learning*, pages 3540–3549. PMLR, 2017.

[43] 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.

[44] Divyansh Garg, Joey Hejna, Matthieu Geist, and Stefano Ermon. Extreme q-learning: Maxent rl without entropy. In *International Conference on Learning Representations (ICLR)*, 2023.

[45] Yinan Zheng, Jianxiong Li, Dongjie Yu, Yujie Yang, Shengbo Eben Li, Xianyuan Zhan, and Jingjing Liu. Safe offline reinforcement learning with feasibility-guided diffusion model. In *The Twelfth International Conference on Learning Representations*, 2024.

[46] Ruiming Liang, Yinan Zheng, Kexin Zheng, Tianyi Tan, Jianxiong Li, Liyuan Mao, Zhihao Wang, Guang Chen, Hangjun Ye, Jingjing Liu, Jinqiao Wang, and Xianyuan Zhan. Dichotomous diffusion policy optimization. In *International Conference on Learning Representations*, 2026.

[47] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

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

[49] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[50] Kexin Zheng, Lauriane Teyssier, Yinan Zheng, Yu Luo, and Xianyuan Zhan. Towards robust zero-shot reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[51] Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy. In *International Conference on Machine Learning*, pages 41837–41853. PMLR, 2025.

[52] Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *International Conference on Machine Learning (ICML)*, 2025.

[53] Liyuan Mao, Haoran Xu, Weinan Zhang, Xianyuan Zhan, and Amy Zhang. Diffusion-dice: In-sample diffusion guidance for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2024.

[54] Sergey Levine, Aviral Kumar, G. Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *ArXiv*, abs/2005.01643, 2020.

[55] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, 2018.

[56] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, 2019.

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

[58] Jianxiong Li, Xiao Hu, Haoran Xu, Jingjing Liu, Xianyuan Zhan, and Ya-Qin Zhang. Proto: Iterative policy regularized offline-to-online reinforcement learning, 2023.

[59] Jianxiong Li, Xianyuan Zhan, Haoran Xu, Xiangyu Zhu, Jingjing Liu, and Ya-Qin Zhang. When data geometry meets deep function: Generalizing offline reinforcement learning. In *International Conference on Learning Representations*, 2023.

[60] Peng Cheng, Xianyuan Zhan, Zhihao Wu, Wenjia Zhang, Shoucheng Song, Han Wang, Youfang Lin, and Li Jiang. Look beneath the surface: Exploiting fundamental symmetry for sample-efficient offline rl. In *Advances in Neural Information Processing Systems*, 2023.

[61] Ilya Kostrikov, Jonathan Tompson, Rob Fergus, and Ofir Nachum. Offline reinforcement learning with fisher divergence critic regularization. In *Proceedings of the 39th International Conference on Machine Learning*, Proceedings of Machine Learning Research, 2021.

[62] Haoyi Niu, Shubham Sharma, Yiwen Qiu, Ming Li, Guyue Zhou, Jianming HU, and Xianyuan Zhan. When to trust your simulator: Dynamics-aware hybrid offline-and-online reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

[63] Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. In *Advances in neural information processing systems*, 2022.

[64] Haoran Xu, Xianyuan Zhan, and Xiangyu Zhu. Constraints penalized q-learning for safe offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[65] Xiangsen Wang, Haoran Xu, Yinan Zheng, and Xianyuan Zhan. Offline multi-agent reinforcement learning with implicit global-to-local value regularization. In *Advances in Neural Information Processing Systems*, 2023.

[66] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on International Conference on Machine Learning*, 2015.

[67] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, 2020.

[68] Sungbin Lim, Eunbi Yoon, Taehyun Byun, Taewon Kang, Seungwoo Kim, Kyungjae Lee, and Sungjoon Choi. Score-based generative modeling through stochastic evolution equations in hilbert spaces. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, 2023.

[69] Yaron Lipman, Marton Havasi, Peter Holderrieth, Neta Shaul, Matt Le, Brian Karrer, Ricky T. Q. Chen, David Lopez-Paz, Heli Ben-Hamu, and Itai Gat. Flow matching guide and code, 2024.

[70] Zhendong Wang, Jonathan J. Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

[71] Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies, 2023.

[72] Huayu Chen, Cheng Lu, Chengyang Ying, Hang Su, and Jun Zhu. Offline reinforcement learning via high-fidelity generative behavior modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[73] 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 *Proceedings of the 40th International Conference on Machine Learning*. JMLR.org, 2023.

[74] Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Diffusion guidance is a controllable policy improvement operator, 2025.

[75] Nicolas Espinosa-Dice, Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kiante Brantley, and Wen Sun. Scaling offline rl via efficient and expressive shortcut models. *Neural Information Processing Symposium (NeurIPS)*, 2025.

[76] Qiyang Li and Sergey Levine. Q-learning with adjoint matching. *arXiv preprint arXiv:2601.14234*, 2026.

[77] Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a vla that learns from experience, 2025.

[78] Yinan Zheng, Tianyi Tan, Bin Huang, Enguang Liu, Ruiming Liang, Jianlin Zhang, Jianwei Cui, Guang Chen, Kun Ma, Hangjun Ye, Long Chen, Ya-Qin Zhang, Xianyuan Zhan, and Jingjing Liu. Unleashing the potential of diffusion models for end-to-end autonomous driving. *arXiv preprint*, 2026.

[79] Jesse Farebrother, Jordi Orbay, Quan Vuong, Adrien Ali Taiga, Yevgen Chebotar, Ted Xiao, Alex Irpan, Sergey Levine, Pablo Samuel Castro, Aleksandra Faust, Aviral Kumar, and Rishabh Agarwal. Stop regressing: Training value functions via classification for scalable deep RL. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 13049–13071. PMLR, 21–27 Jul 2024.

[80] Xiaoteng Ma, Junyao Chen, Li Xia, Jun Yang, Qianchuan Zhao, and Zhengyuan Zhou. Dsac: Distributional soft actor-critic for risk-sensitive reinforcement learning. *Journal of Artificial Intelligence Research*, 83, 2025.

[81] Yevgen Chebotar, Quan Vuong, Karol Hausman, Fei Xia, Yao Lu, Alex Irpan, Aviral Kumar, Tianhe Yu, Alexander Herzog, Karl Pertsch, et al. Q-transformer: Scalable offline reinforcement learning via autoregressive q-functions. In *Conference on Robot Learning*, pages 3909–3928. PMLR, 2023.

[82] Jost Tobias Springenberg, Abbas Abdolmaleki, Jingwei Zhang, Oliver Groth, Michael Bloesch, Thomas Lampe, Philemon Brakel, Sarah Maria Elisabeth Bechtle, Steven Kapturowski, Roland Hafner, Nicolas Heess, and Martin Riedmiller. Offline actor-critic reinforcement learning scales to large models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 46323–46350. PMLR, 21–27 Jul 2024.

[83] Johan Samir Obando Ceron, Ghada Sokar, Timon Willi, Clare Lyle, Jesse Farebrother, Jakob Nicolaus Foerster, Gintare Karolina Dziugaite, Doina Precup, and Pablo Samuel Castro. Mixtures of experts unlock parameter scaling for deep RL. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 38520–38540. PMLR, 21–27 Jul 2024.

[84] Peter Dayan and Geoffrey E Hinton. Feudal reinforcement learning. *Advances in neural information processing systems*, 5, 1992.

[85] Thomas G Dietterich. Hierarchical reinforcement learning with the maxq value function decomposition. *Journal of artificial intelligence research*, 13:227–303, 2000.

[86] Alexander Vezhnevets, Volodymyr Mnih, Simon Osindero, Alex Graves, Oriol Vinyals, John Agapiou, et al. Strategic attentive writer for learning macro-actions. *Advances in neural information processing systems*, 29, 2016.

[87] Shubham Pateria, Budhitama Subagdja, Ah-hwee Tan, and Chai Quek. Hierarchical reinforcement learning: A comprehensive survey. *ACM Computing Surveys (CSUR)*, 54(5):1–35, 2021.

[88] Tejas D Kulkarni, Karthik Narasimhan, Ardavan Saeedi, and Josh Tenenbaum. Hierarchical deep reinforcement learning: Integrating temporal abstraction and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

[89] Kevin Zakka, Yuval Tassa, and MuJoCo Menagerie Contributors. MuJoCo Menagerie: A collection of high-quality simulation models for MuJoCo, 2022. URL http://github.com/google-deepmind/mujoco_menagerie.

[90] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[91] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[92] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

[93] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[94] Kwanyoung Park, Seohong Park, Youngwoon Lee, and Sergey Levine. Scalable offline model-based rl with action chunks. In *International Conference on Learning Representations*, 2026.

A LLM Usage

In this paper, we employed Large Language Models (LLMs) solely for polishing the writing. No parts of the technical content, experimental results, or conclusions were generated by LLMs.

B Theoretical Proofs

B.1 Proof of Theorem 1 (Temporal Compositionality)

Proof. By definition, $G(s, s_k, k) = \mathbb{E}_\pi \left[\sum_{t=0}^{k-1} \gamma^t r_t \mid s_0 = s\right]$, where k denotes the horizon between states s and s' . For any intermediate state s_i reached at time $i < k$, we can partition the summation:

$$G(s, s_k, k) = \mathbb{E}_\pi \left[\sum_{t=0}^{i-1} \gamma^t r_t + \sum_{t=i}^{k-1} \gamma^t r_t \;\middle|\; s_0 = s, (s_k, k)\right] \tag{21}$$

$$= \mathbb{E}_{i\sim\text{Unif}\{1,\dots,k-1\}, s_i\sim\pi} \left(\mathbb{E}_\pi \left[\sum_{t=0}^{i-1} \gamma^t r_t \;\middle|\; s_0 = s, (s_i, i)\right] \right. \tag{22}$$

$$\left. + \mathbb{E}_\pi \left[\gamma^i \sum_{t=0}^{k-i-1} \gamma^t r_{t+i} \;\middle|\; s_0 = s_i, (s_k, k-i)\right]\right) \tag{23}$$

$$= \mathbb{E}_{i\sim\text{Unif}\{1,\dots,k-1\}, s_i\sim\pi} \left[G(s, s_i, i) + \gamma^i G(s_i, s_k, k-i)\right], \tag{24}$$

where π denotes the behavioral distribution. Under the function approximation $\hat{G}$, let the error at each compositional step be bounded by ϵ_{comp}. For a trajectory composed of M segments $(s_0, s_1, \dots, s_M)$, the approximated return is equivalent to $\sum_{t=0}^{M-1} \gamma^t \hat{G}(s_t, s_{t+1}, 1)$, which implies the accumulated error is at most $\sum_{t=0}^{M-1} \gamma^t \epsilon_{\text{comp}} \le M\epsilon_{\text{comp}}$, under a linear growth w.r.t segments. ■

B.2 Proof of Theorem 2 (Contraction Mapping)

Proof. We evaluate the supremum norm between the operator applied to two arbitrary bounded value functions V_1 and V_2:

$$\begin{aligned}
\|\mathcal{T}_{\text{stitch}} V_1 - \mathcal{T}_{\text{stitch}} V_2\|_\infty &= \max_s \left| \max_{k_1\in\{1,\dots,K\}} \left[G(s, s_{k_1}, k_1) + \gamma^{k_1} V_1(s_{k_1})\right] \right. \\
&\qquad \left. - \max_{k_2\in\{1,\dots,K\}} \left[G(s, s_{k_2}, k_2) + \gamma^{k_2} V_2(s_{k_2})\right] \right| \\
&\le \max_s \left| \max_{k_1\in\{1,\dots,K\}} \left[G(s, s_{k_1}, k_1) + \gamma^{k_1} V_1(s_{k_1})\right] - \left(G(s, s_{k_1}, k_1) + \gamma^{k_1} V_2(s_{k_1})\right) \right| \\
&\le \max_s \max_{k\in\{1,\dots,K\}} \left| \left(G(s, s_k, k) + \gamma^k V_1(s_k)\right) - \left(G(s, s_k, k) + \gamma^k V_2(s_k)\right) \right| \\
&= \max_s \max_{k\in\{1,\dots,K\}} \gamma^k \left|V_1(s_k) - V_2(s_k)\right| \\
&\le \gamma^{\tilde{k}} \|V_1 - V_2\|_\infty \\
&\le \gamma \|V_1 - V_2\|_\infty.
\end{aligned} \tag{25}$$

The last inequality holds as any valid temporal transition requires at least one environment step, $\tilde{k} \ge 1$, which implies $\gamma^{\tilde{k}} \le \gamma < 1$, proving that $\mathcal{T}_{\text{stitch}}$ is a strict γ-contraction. By the Banach Fixed-Point Theorem, repeated application of $\mathcal{T}_{\text{stitch}}$ converges to a unique fixed point V^*. ■

C Details on the Toy Example

The values visualized in Figure 2 are computed *analytically*: for each value-learning scheme, we apply exact dynamic programming over the transitions contained in the dataset (the green and purple

paths), proceeding recursively from the goal cell at the top-right corner back to the start cell at the bottom-left corner. We use $\gamma = 0.99$ and initialize the value of the goal cell to $V((3,3)) = 0$. For example, under one-step TD, $Q((2,3), \text{right}) = 1 + \gamma \cdot V((3,3)) = 1$. Under this setup, there exist unique optimal path passes through $(1,0)$. The key failure mode of fixed-horizon modeling is caused by the negative reward at $(1,2)$ that distorts the value of nearby states along the purple path, which further avoid the optimal transition $(0,0) \rightarrow (1,0)$. Figure 2 shades only the cells that are critical for exposing this failure mode.

D Experimental Setup

D.1 Environment

We evaluate *VAST* on the OGBench task suite [31]. OGBench is a benchmark for goal-conditioned offline RL, with challenging robotic locomotion and manipulation tasks, including whole-body humanoid control, maze navigation, and object manipulation. Our results cover 30 tasks across 6 widely used domains (`scene-play`, `cube-single-play`, `cube-double-play`, `puzzle-4x4-play`, `antmaze-large-navigate`, `antmaze-giant-navigate`), denoted as *Basic* (refer to Figure 7), and 20 tasks across 4 highly challenging domains (`cube-triple-play`, `cube-quadruple-play`, `puzzle-4x5-play`, `puzzle-4x6-play`), denoted as *Hard* (refer to Figure 8), for a total of 50 tasks on OGBench in `singletask` variant for offline RL. All tasks we evaluated are state-based; evaluation is performed on five pre-defined state-goal pairs. We introduce their settings in detail as follows:

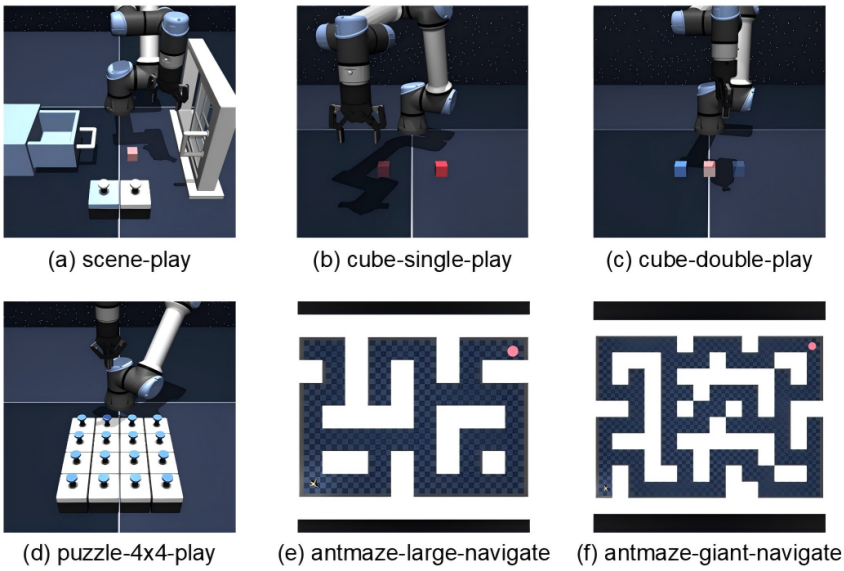

Figure 7: **OGBench Basic.** We experiment on 6 majorly used domains on OGBench [31] in singletask variant, shown as (a)-(f). Each domain contains 5 tasks.

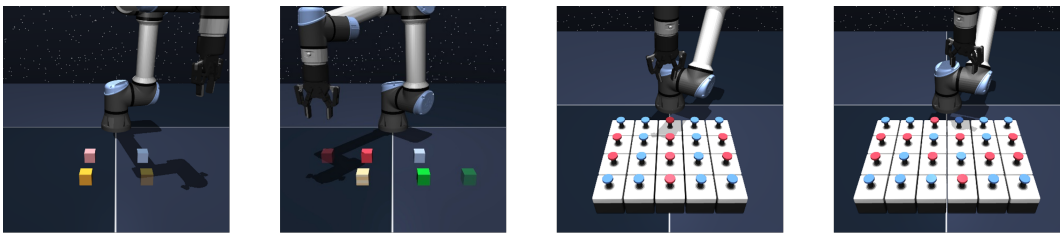

Figure 8: **OGBench Hard.** We experiment on 4 highly challenging domains on OGBench [31] in singletask variant, shown as (a)-(d). Each domain contains 5 tasks.

Manipulation. Manipulation tasks are performed with the UR5e robot arm and Robotiq 2F-85 gripper models from MuJoCo Menagerie [89] and the drawer, window, and button box models from Meta-World [90]. We utilize `play` datasets for all domains, collected by open-loop, non-Markovian scripted policies defined by Park et al. [31].

- **scene**: Long-horizon manipulation involves interacting with multiple object types within one environment, including a cube block, a window, a drawer, and two button locks, where actions change object states and enable subsequent manipulation.
- **cube**: Pick-and-place manipulation of cube-shaped blocks to form specified target configurations. We use `single` to move one cube and `double` to arrange two cubes in the shared domain for common evaluation. We further select `triple` and `quadruple`, which place three and four cubes, respectively, as the challenge domains, thereby strongly increasing the combinatorial difficulty of assembling target configurations.
- **puzzle**: Solving with a robot arm where pressing buttons induces deterministic state transitions according to the puzzle rule (e.g., toggling the pressed cell and its neighbors). This series of domains is designed to test the agent's combinatorial generalization abilities. We use `4x4` (4×4 grid) for common evaluation. We further select hard settings with larger grid sizes, `4x5` (4×5 grid) and `4x6` (4×6 grid), as the challenging domains.

Navigation. Environments are adopted from Mujoco [91], and the success of this series of domains is determined by the achieved proximity to a specified goal location in the maze. We utilize `navigate` datasets for all domains, collected by a noisy expert policy that navigates the maze by repeatedly reaching randomly sampled goals.

- **antmaze**: Controlling a quadrupedal Ant agent with 8 degrees of freedom (DoF) to reach a specified goal location, with success evaluated by the achieved proximity to the goal in the maze. We select `large` maze and `giant` maze (twice the size of `large`) in this work.

D.2 Evaluation

We evaluate all methods trained on standard datasets provided by OGBench, and report the success rate at 1M gradient steps for each task. Each evaluation consists of 50 episodes. The data specifications are shown in Table 2.

Table 2: **Data specifications.** Reported from Park et al. [31]

Dataset	Transitions	Episodes	Data Episode Length	State Dim.	Action Dim.
`scene-play`	1M	1000	1000	40	5
`cube-single-play`	1M	1000	1000	28	5
`cube-double-play`	1M	1000	1000	37	5
`puzzle-4x4-play`	1M	1000	1000	83	5
`antmaze-large-navigate`	1M	1000	1000	29	8
`antmaze-giant-navigate`	1M	500	2000	29	8
`cube-triple-play`	3M	3000	1000	46	5
`cube-quadruple-play`	5M	5000	1000	55	5
`puzzle-4x5-play`	3M	3000	1000	99	5
`puzzle-4x6-play`	5M	5000	1000	115	5

D.3 Baselines

In this section, we provide the details of the baselines in this work. We select *IQL*, *FQL*, *IFQL*, *FQL-n*, *IFQL-n*, *QC*, *VALUE-FLOWS*, and *FLOQ*, in total of 8 baselines for a comprehensive comparison. Specifically, we report the performance of *IQL* from Park et al. [52] and use the open-source codebases for experiments on others, using the corresponding default hyperparameters.

- ***IQL* [35].** *IQL* is a strong and stable Gaussian-policy offline RL baseline. The key idea of *IQL* is to learn a state value function via upper expectile regression over in-dataset action values, and then use this value estimate to bootstrap the Q-function. The policy is subsequently extracted via

AWR-style [43] advantage-weighted behavioral cloning, thereby enabling policy improvement while remaining close to the data distribution.

- ***FQL/FQL-n [52].*** *FQL* is a strong flow-policy offline RL baseline built on a behavior-regularized actor-critic framework. The key idea of *FQL* is to train a behavior-cloned flow model for expressive action modeling, while learning a separate one-step policy that maximizes the critic under a distillation regularizer. Specifically, *FQL* avoids unstable backpropagation through time and eliminates costly iterative inference, while still retaining much of the expressivity of flow-based policies by decoupling value maximization from the iterative flow-generation process. *FQL-n* is the n-step backup variant of *FQL*.
- ***IFQL/IFQL-n [52].*** *IFQL* is a flow-based variant of *IQL* [35] that combines expectile-regression value learning with an imitation-trained flow policy and rejection sampling for policy extraction. *IFQL*, therefore, is a traditional, strong, and stable in-sample baseline that benefits from expressive flow-based action modeling while avoiding direct optimization through the generative process. *IFQL-n* is the n-step backup variant of *IFQL*.
- ***QC [17].*** *QC* is a strong baseline designed for long-horizon sparse-reward problems via action chunking. *QC* learns a flow-matching behavior policy over temporally extended action chunks and a chunk-level critic, and performs rejection sampling to choose the action sequence with the highest predicted value. This chunked formulation enables longer-horizon TD back-ups and more temporally coherent exploration, which substantially improves learning in tasks where single-step control is ineffective. It is a representative baseline that explicitly addresses long-horizon difficulty through temporal abstraction and chunk-level value learning.
- ***VALUE-FLOWS [21].*** *VALUE-FLOWS* is a strong distributional RL baseline that models the full return distribution using flow matching, rather than collapsing future outcomes into a single scalar Q-value. It learns a return vector field that satisfies the distributional Bellman equation and additionally uses estimated return variance to reweight training toward more uncertain transitions. *VALUE-FLOWS* is particularly appealing for long-horizon tasks with multimodal or uncertain returns.
- ***FLOQ [20].*** *FLOQ* is a strong value-learning baseline that scales value estimation through iterative computation. Specifically, *FLOQ* represents the Q-function as a flow-matching velocity field that transforms noise into a scalar Q-value through numerical integration, with dense supervision applied across intermediate integration steps. In practice, *FLOQ* is instantiated on top of *FQL*, preserving the expressive flow-policy framework while substantially improving critic capacity and enabling a form of test-time compute scaling through additional integration steps. It is a particularly strong baseline for evaluating improvements from value scaling.

E Implementation

This section provides additional implementation information. See pseudo-code in Section E.3, implementation and structural details in Section E.1, and hyperparameters in Section E.2.

E.1 Implementation details

We implement *VAST* in JAX [92] on top of recent study by Park et al. [13].

Architectures. We train 5 neural networks in parallel: 2 of flow policy networks (stitching policy π_{stitch}; execute policy π_{exec}), and 3 of value networks (state value V; state-action value Q; horizon-based cumulative return G). We leverage the multi-layer perceptron (MLP) for each network; the depth and width are held constant within each domain.

Value learning. In all experiments, we learn the optimal state-value function V using IQL-style expectile regression [35]. The sampled bootstrap target $(k, s_k) \sim \pi_{\text{stitch}}(\cdot \mid s)$ could enable a greedier and policy-aware value update. To better align value learning with both the *stitching* and *execution* policies, we approximate V^{π}_{target} in Eq. (17) as

$$V^{\pi}_{\text{target}} = G(s, s_k, k) + \gamma^k Q\big(s_k, \pi_{\text{exec}}(a_k \mid s_k, s_k^+)\big), \tag{26}$$

where $(k^+, s_k^+) \sim \pi_{\text{stitch}}(\cdot \mid s_k)$. The sampled horizon k is the continuous output of the flow policy π_{stitch} and enters G and γ^k directly. Here, π_{exec} is trained to produce actions within the data

distribution. As a result, the second term provides a stable approximation to the continuation value at s_k, while still reflecting the behavior induced by the combined stitching-and-execution policy.

For learning the G-function, given a batch of states $\{s_t\}$, we first uniformly sample a future state s_{t+k} from the same trajectory, where the horizon k is sampled from $\{1, \ldots, \min(K, t_{\text{terminal}} - t)\}$ with t_{terminal} the terminal step of the trajectory, and K denotes the maximum horizon. We then uniformly sample an intermediate state s_{t+i} between s_t and s_{t+k}, with $i \sim \{1, \ldots, k-1\}$; only samples with $k \geq 2$ enter the compositional objective. The G-function is trained with the Monte Carlo objective in Eq. (8) and the compositional objective in Eq. (9), using a fixed balance weight of 0.5. The maximum horizon K controls the temporal range of MC supervision for the G-function. In practice, choosing K based on the domain helps balance the expressivity of the G-network against task complexity. Detailed domain-specific settings are provided in Table 4.

Policy extraction. We use weighted regression for the objective of *stitching* policy, which has been fully studied in previous works [45, 50, 46]. It starts with a regularized optimization problem:

$$\max_{\pi_{\text{stitch}}} \mathbb{E}_{s\sim\mathcal{D}}\left[\mathbb{E}_{\pi_{\text{stitch}}}\left[V^{\pi}_{\text{target}} - V(s)\right] - \frac{1}{\beta} D_{\text{KL}}(\pi_{\text{stitch}}(\cdot|s) \,\|\, \mu(\cdot|s))\right], \tag{27}$$

where μ denotes the data distribution. Following the derivation of Zheng et al. [45], we could have the closed-form solution:

$$\pi^{\star}_{\text{stitch}}(k, s_k \mid s) \propto \mu(k, s_k \mid s) \cdot \exp\left(\beta \cdot \left(V^{\mathcal{D}}_{\text{target}} - V(s)\right)\right), \tag{28}$$

where $V^{\mathcal{D}}_{\text{target}} = G(s_t, s_{t+k}, k) + \gamma^k Q(s_{t+k}, a_{t+k})$, with $(k, s_{t+k}, a_{t+k}) \sim \mathcal{D}$. The above form gives us the weighted objectives for the stitching flow networks as Eq. (19), follow the formulation by Zheng et al. [45]. Given the parameterized execute flow network $\epsilon_{\text{execute}}$, the objective is:

$$\mathcal{L}_{\epsilon_{\text{execute}}} = \mathbb{E}_{t\sim U[0,1], \epsilon\sim\mathcal{N}(\mathbf{0},\mathbf{I}), (s,a,s_{\text{goal}})\sim\mathcal{D}}\left[\left\| \epsilon - \epsilon_{\text{execute}}\left(a^t, s, s_{\text{goal}}, t\right)\right\|^2\right]. \tag{29}$$

At inference time, we use rejection sampling to select the output of the execute policy π_{exec} with N_{rej} variants by tasks. See Section E.2 for detailed hyperparameters.

Goal sampling mechanism for π_{exec}. π_{exec} is trained in a goal-conditioned behavior-cloning (BC) manner, following the setting of Park et al. [13]. We denote the state-action-goal sampling distribution as $p_{\mathcal{D}}$, while $p_{\mathcal{D}}(g \mid s, a)$ is the distribution from which the conditioning goal s_{goal} in Eq. (29) is drawn. $p_{\mathcal{D}}$ is a mixture of three components: $p^{\mathcal{D}}_{\text{cur}}$, the Dirac delta at the current state s; $p^{\mathcal{D}}_{\text{traj}}$, the uniform distribution over the future states of the same trajectory; and $p^{\mathcal{D}}_{\text{rand}}$, the uniform distribution over all dataset states. The ratios of these distributions are tunable hyperparameters that sum to 1, refers to *subgoal sample probability* shown in Table 3. We use *uniform* sampling over all future states up to the end of the trajectory.

E.2 Hyperparameters

In this section, we provide the detailed hyperparameter setup in Table 3 for the general hyperparameters, and Table 4 for the domain-specific hyperparameters.

Table 3: **General hyperparameters**.

	Hyperparameter	
General	Optimizer	Adam
	Learning rate	3e-4
	Offline learning steps	1,000,000
	Mini-batch	512
	Soft update factor	0.005
	Flow steps T	10
	Gradient clip	1
	Expectile factor τ	0.9
	Clip Q	true (`antmaze-giant`), false (others)
	Subgoal sample probability $[p^{\mathcal{D}}_{\text{cur}}, p^{\mathcal{D}}_{\text{traj}}, p^{\mathcal{D}}_{\text{rand}}]$	[0,1,0] (`antmaze-large/giant`), [0.1, 0.8, 0.1] (others)
Architecture	MLP hidden dimension	[512, 512, 512, 512]
	Activation function	GELU [93]
	Layernorm	true

Table 4: **Domain-specific hyperparameters.**

Domain	K	β	N_{rej}	γ	λ
`scene-play`	50	0.03	4	0.995	0.25
`cube-single-play`	50	0.08	4	0.99	0.5
`cube-double-play`	50	0.08	4	0.99	0.5
`puzzle-4x4-play`	10	0.01	4 (task 2,5); 16 (others)	0.995	0.5
`antmaze-large-navigate`	50	0.05	4	0.995	0.5
`antmaze-giant-navigate`	50	0.05	4	0.995	0.5
`cube-triple-play`	25	0.001	4	0.999	0.5
`cube-quadruple-play`	10	0.001	16	0.999	0.1
`puzzle-4x5-play`	10	0.005	32	0.995	0.5
`puzzle-4x6-play`	10	0.005	32	0.995	0.5

E.3 Algorithm Pseudo-code

We provide pseudo-codes for *VAST* as Algorithm 1.

Algorithm 1 *VAST*

// Train time:
Initialize G, V, Q, π_{stitch}, π_{exec}
while not converged **do**
  Sample batch $(s_t, a_t, r_t, s_{t+1}, s_{\text{goal}}, s_k, a_k, k, s_i, i) \sim D$ from offline data
  // Policy Learning
  Update stitching policy $\pi_{\text{stitch}}(s_k, k \mid s)$ with loss (Eq. (19))
  Update execute policy $\pi_{\text{exec}}(a \mid s, s_{\text{goal}})$ with loss (Eq. (29))
  // Value Learning
  Update G with loss (Eq. (10))
  Update V with loss (Eq. (17))
  Update Q with loss (Eq. (15))
end while

// Test time:
Get initial state s
while not done **do**
  // Sub-goal prediction
  Sample s_{goal} from $\pi_{\text{stitch}}(\cdot|s)$
  // Execution
  Sample $a_1, a_2, ..., a_N$ from $\pi_{\text{exec}}(\cdot|s, s_{\text{goal}})$
  Set $a \leftarrow \arg\max_{a_1,a_2,...,a_N} Q(s, a_i)$
  Update s
end while

F Additional Results

F.1 Full Results

In this section, we provide the full experimental results on OGBench (Table 5). Scores are aggregated over 8 random seeds, with $\pm$ indicating standard deviations. Best performances are highlighted with color. See Section D.2 for the evaluation portal.

Table 5: **Results on OGBench.**

		1-step TD			n-step TD		Chunking	Generative		Stitching
	Task	IQL	FQL	IFQL	FQL-n	IFQL-n	QC	VALUE-FLOWS	FLOQ	VAST
scene-play	1	94	100±0	99±1	100±0	99±1	100±0	100±1	100±0	100±0
	2	12	81±14	55±26	26±15	0±0	99±2	99±1	82±11	98±3
	3	32	97±3	76±9	85±11	59±4	99±1	95±6	98±2	97±3
	4	0	9±7	0±0	0±0	0±1	1±2	5±8	4±4	8±4
	5	0	0±0	0±0	0±0	0±0	0±0	0±0	0±0	1±2
	agg.	28	57±7	46±12	42±8	32±2	60±1	60±5	57±5	61±3
cube-single-play	1	88	96±7	84±7	97±2	77±8	98±2	97±2	99±1	99±2
	2	85	97±3	89±5	96±1	86±5	94±4	97±4	98±3	100±1
	3	91	98±3	92±3	99±1	87±6	97±4	99±1	99±2	100±1
	4	73	96±5	87±4	98±2	78±6	89±4	95±3	96±3	98±3
	5	78	96±3	84±7	97±5	84±5	89±9	86±8	89±9	94±4
	agg.	83	97±4	87±5	97±3	82±6	93±5	95±4	96±5	98±2
cube-double-play	1	27	67±9	32±6	16±6	9±4	85±6	96±5	64±17	93±2
	2	1	38±10	11±4	1±1	7±5	78±6	66±11	61±22	87±4
	3	0	23±6	4±3	1±1	4±3	74±5	63±15	57±12	85±9
	4	0	5±2	1±2	0±0	0±0	26±9	20±11	12±9	22±7
	5	4	17±9	10±6	2±1	2±2	72±9	61±9	52±15	89±7
	agg.	7	30±8	12±5	4±3	4±3	67±7	61±11	49±16	75±6
puzzle-4x4-play	1	12	29±6	41±12	42±4	17±6	67±11	18±12	61±8	69±11
	2	7	14±7	17±9	20±3	9±4	1±2	33±2	22±8	27±10
	3	9	18±6	48±8	35±8	15±3	64±17	18±12	40±15	81±11
	4	5	9±2	20±6	20±5	10±3	10±8	20±5	19±6	33±10
	5	4	6±5	7±3	17±7	7±4	3±3	14±6	14±4	21±5
	agg.	7	15±5	27±8	27±6	12±4	29±10	21±8	31±9	46±10
antmaze-large-navigate	1	48	87±6	34±20	80±10	15±10	4±5	22±11	96±3	87±7
	2	42	63±10	17±7	59±14	0±0	0±0	53±14	83±11	74±2
	3	72	96±3	49±12	91±5	57±8	19±10	53±23	98±2	81±8
	4	51	80±8	15±13	83±7	8±6	0±0	50±17	92±7	80±10
	5	54	85±6	44±26	86±6	56±7	8±3	61±5	95±3	84±2
	agg.	53	82±7	32±17	80±9	27±7	6±5	48±15	93±6	81±7
antmaze-giant-navigate	1	0	12±12	0±0	3±6	0±0	0±0	0±0	75±12	1±1
	2	1	21±18	0±0	88±3	0±0	0±0	1±1	37±30	46±7
	3	0	3±1	0±0	17±7	0±0	0±0	0±0	0±0	14±6
	4	0	10±21	0±0	40±32	0±0	0±0	0±0	17±28	7±5
	5	19	7±18	1±4	36±13	1±1	0±0	13±7	15±31	33±7
	agg.	4	11±16	0±2	37±16	0±0	0±0	3±3	29±24	20±6
cube-triple-play	1	-	6±8	3±2	1±1	1±1	19±11	53±22	17±10	61±17
	2	-	0±0	0±0	0±0	0±0	2±2	0±0	0±0	10±4
	3	-	0±0	0±0	0±0	0±0	1±1	8±5	1±2	10±6
	4	-	0±0	0±0	0±0	0±0	0±0	1±2	0±0	2±2
	5	-	0±0	0±0	0±0	0±0	0±0	1±1	0±0	2±2
	agg.	-	1±4	1±1	0±0	0±0	4±5	13±10	4±5	17±8
cube-quadruple-play	1	-	0±0	0±0	0±0	0±0	1±1	0±0	0±0	5±3
	2	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	3	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	4	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	5	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	agg.	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	1±1
puzzle-4x5-play	1	-	75±7	78±15	92±3	69±9	100±0	59±9	67±7	100±1
	2	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	3	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	4	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	5	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	agg.	-	15±3	16±7	18±1	14±4	20±0	12±4	13±3	20±0
puzzle-4x6-play	1	-	20±12	0±0	18±19	15±12	21±10	17±13	12±6	38±38
	2	-	0±1	0±0	0±0	1±1	0±0	0±0	1±1	0±0
	3	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	4	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	5	-	0±0	0±0	0±0	0±0	0±0	0±0	0±0	0±0
	agg.	-	4±5	0±0	4±9	3±5	4±4	3±6	3±3	8±17

F.2 Learning Curves

We provide the learning curves on all OGBench tasks. Shaded regions indicate the minimum and maximum across seeds.

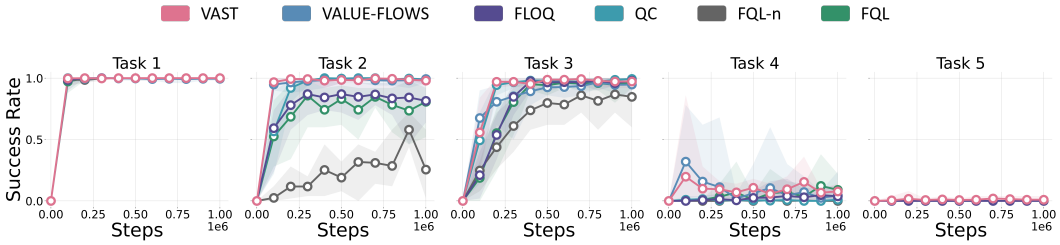

Figure 9: `scene-play`.

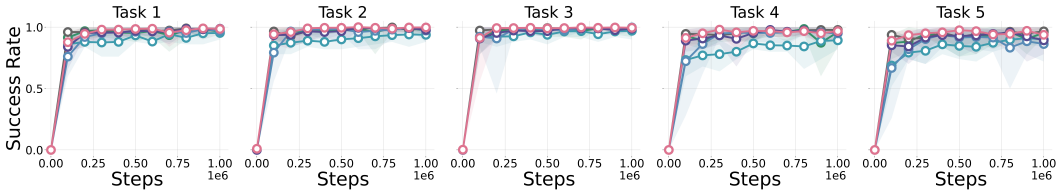

Figure 10: `cube-single-play`.

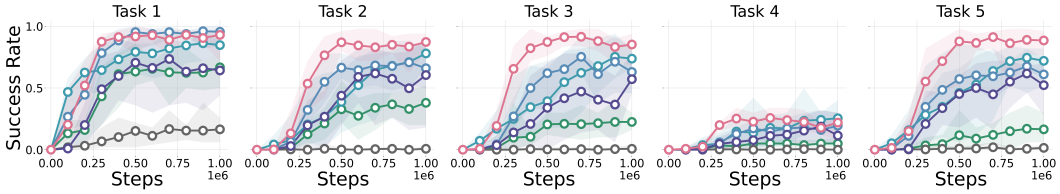

Figure 11: `cube-double-play`.

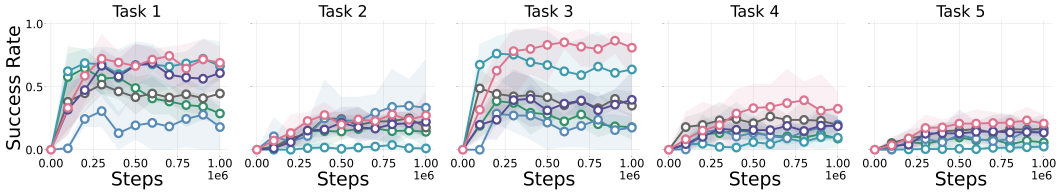

Figure 12: `puzzle-4x4-play`.

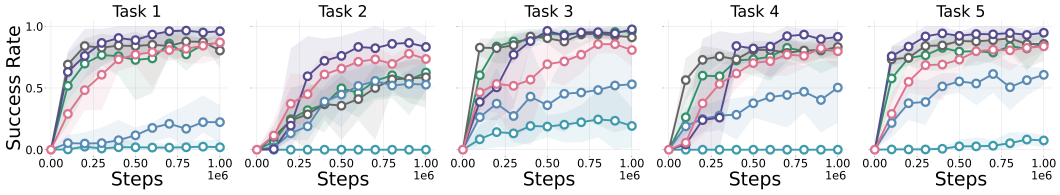

Figure 13: `antmaze-large-navigate`.

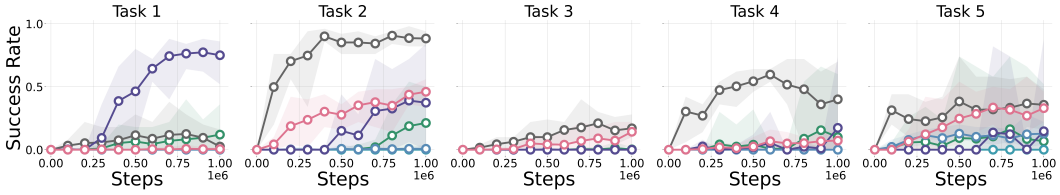

Figure 14: `antmaze-giant-navigate`.

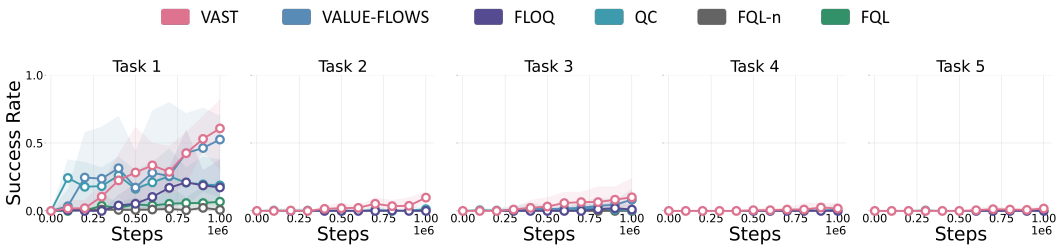

Figure 15: `cube-triple-play`.

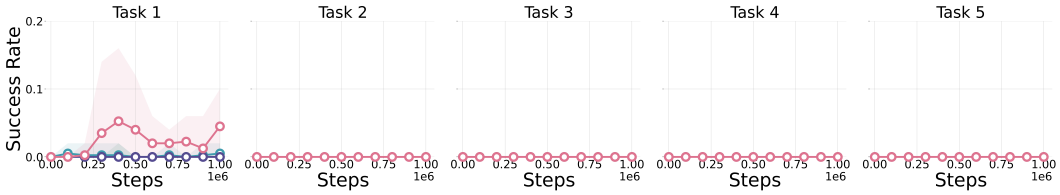

Figure 16: `cube-quadruple-play`.

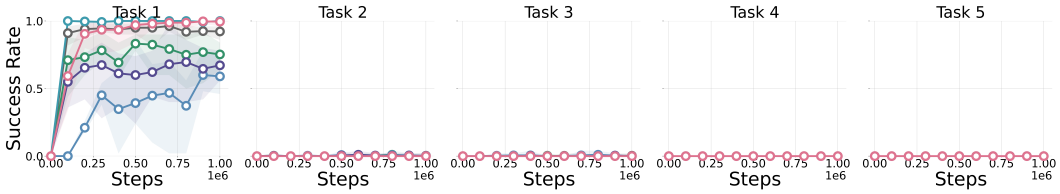

Figure 17: `puzzle-4x5-play`.

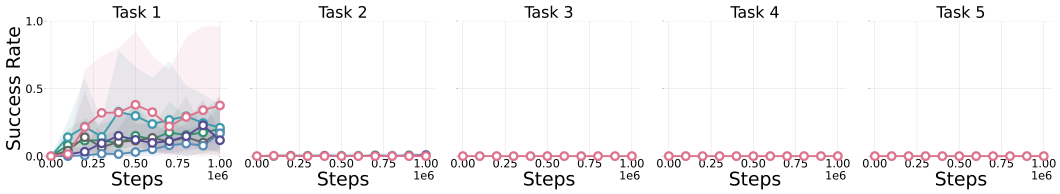

Figure 18: `puzzle-4x6-play`.

G Additional Discussions

This section provides discussions on important design choices: the relation of *VAST* to model-based RL (Appendix G.1), why the G-function must be conditioned on the horizon (Appendix G.2), and the learning complexity of G (Appendix G.3).

G.1 Comparison of *VAST* to model-based RL

Model-based RL shares the motivation of mitigating long-horizon value-estimation error by adopting temporally extended modeling. Park et al. [94] learns an action-chunk dynamics model that predicts future states from action sequences and uses imagined rollouts to assist value learning and policy improvement. While *VAST* is fundamentally model-free that never learns a dynamics model nor generates imagined transitions. Instead, the G-function evaluates the return between *actual* state pairs observed in the dataset, and π_{stitch} optimizes directly over actionable, in-support sub-goals. The predicted sub-goal in *VAST* plays the role of a value-backup anchor and an execution target.

However, this comparison is only the first step, far from the end of the story. We open the question of how to combine a learned dynamics (world) model into a better horizon-adaptive framework that enables a cleaner credit assignment. This question hinges on how to make a learned dynamic or world model, within the rapidly growing field of model-based learning, meaningfully shape what the

policy should improve—so that an aggressive policy optimization process does not rely on imitation-like signals or a patchwork of separate components.

G.2 Why condition G on the Horizon?

Since π_{exec} does not consume the horizon k, one may ask whether k could also be dropped from the G-function. However, horizon-free $G(s, s^+)$ admits infinitely many solutions, because a successor s^+ can be reached from s at an arbitrary horizon. This leads to two problems: (1) increased estimation error in G, and (2) loss of the reachability information that G is supposed to carry. Specifically, consider two trajectories that share the prefix (s_0, a_0, r_0, s_1) and the same suffix leading to a successful terminal state s_{success}:

$$\tau_1 = (s_0, a_0, r_0, s_1, a_1, r_1, s_2, a_2, r_2, s_3, a_3, r_3, s_4, \ldots, s_{\text{success}}), \quad \text{with } s_1 = s_3 = s^+,$$
$$\tau_2 = (s_0, a_0, r_0, s_1, a_1, r_1, s_2, \ldots, s_{\text{success}}), \quad \text{with } s_1 = s^+.$$

Trajectory τ_2 is the shorter and optimal, while τ_1 contains a redundant detour from s_1 back to $s_3 = s^+$. A horizon-free G would be required to satisfy both

$$V(s_0) \leftarrow G(s_0, s^+) + \gamma\, V(s^+) \qquad \text{and} \qquad V(s_0) \leftarrow G(s_0, s^+) + \gamma^3\, V(s^+).$$

A single scalar $G(s_0, s^+)$ cannot account for both the immediate and the delayed arrival at s^+. The rewards accumulated over different horizons are inseparably conflated, and $V(s_0)$ cannot be driven to its optimal value even when $V(s^+)$ is optimal. Moreover, the compositional property of Theorem 1 is no longer well defined, thereby G would also lose the temporal regularization.

G.3 On the learning complexity of G

With the definition of G (Section 3), G is in fact easier to fit than standard value functions. First, $G(s, s^+, k)$ conditions on richer and more controlled information, and regresses onto finite-horizon, grounded Monte-Carlo targets over at most K steps (Eq. (8)), rather than onto bootstrapped infinite-horizon discounted returns. Second, the compositional objective (Eq. (9)) imposes a multi-scale consistency constraint that regularizes G_ψ across horizons. As a result, the estimation error of G is bounded by the finite-horizon target range and does not compound through recursive bootstrapping in the way that one-step TD errors do.

H Limitations and Future Works

H.1 Computation Cost Comparison

The core components introduced by *VAST* include an auxiliary value function and two policies with distinct functional roles. Consequently, with the same network architecture, *VAST* has more parameters than the baselines. We report the parameter comparison in Table 6 and the computation time in Table 7. All methods are evaluated on the `cube-double-play` domain using an NVIDIA A100 GPU, with the layer hidden dimension fixed to 512 and the batch size fixed to 512. Parameter counts include the target networks, and computation time is measured over 1M training steps. Under this controlled setting, the training time of *VAST* is comparable to that of *FLOQ*, remaining at a moderate computational cost. We consider this overhead a reasonable trade-off for the improved flexibility and optimality that *VAST* offers.

Table 6: Total parameters.

Methods	Parameter Count (M)
VAST	6.6
FQL	4.9
QC	5.0
VALUE-FLOWS	4.9
FLOQ	4.5

Table 7: Computation time per 1M steps.

Methods	Computation Time (H)
VAST	1.7
FQL	0.6
QC	2.1
VALUE-FLOWS	4.9
FLOQ	1.7

H.2 Scalability to High-Dimensional Observations

In the current instantiation, π_{stitch} predicts s_k directly in the raw state space, and G is conditioned on raw state pairs. All tasks considered in this work are state-based with moderate dimensionality (Table 2). For high-dimensional states or pixel-based observations, directly generating sub-goals in the observation space would increase the expressivity burden on π_{stitch} and reduce training efficiency. However, *VAST* can be readily combined with standard representation-learning techniques to maintain practical efficiency (e.g., an encoder-decoder architecture that maps raw observations into a low-dimensional latent space). Future work will make a validation on pixel-based setting.

H.3 Greedier Diffusion Policy Optimization

In this work, we employ a diffusion policy extraction method that is more stable, albeit conservative. Meanwhile, recent studies have proposed more aggressive optimization procedures for diffusion policies, including Liang et al. [46] and Li and Levine [76]. Future work will investigate enhanced policy learning strategies to improve robustness and controllability.

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 [N/A] 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 [N/A] 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 Appendix H.

   Guidelines:

   - The answer [N/A] 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 Appendix B.

Guidelines:

- The answer [N/A] 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, experimental settings, hyperparameters, and baseline details are included in Appendix D-E.

Guidelines:

- The answer [N/A] 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 [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://neurips.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://neurips.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) necessary to understand the results?

   Answer: [Yes]

   Justification: Yes, all hyperparameters and experimental setup details necessary to understand the results are provided in Appendix D-E.

   Guidelines:

   - The answer [N/A] 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 algorithms, several random seeds are used. All results are presented as mean ± standard deviation.

   Guidelines:

   - The answer [N/A] 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, Appendix H provides detailed information about the specific computational demand used for the experiments.

   Guidelines:

   - The answer [N/A] 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 to preserve anonymity.

   Guidelines:

   - The answer [N/A] 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 main paper discusses both potential positive impacts and limitations of the work.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] 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., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer [N/A] 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 Appendix D, with licenses and terms of use clearly indicated.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] 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: [N/A]

    Justification: The paper does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] 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.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

    Justification: Yes, see Appendix A.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.