Faster Policy Learning from Multiple Experts: Stitch, Imitate, Optimize

Braham Snyder[1], **Jian Qian**[2], **Chen-Yu Wei**[1]
braham.snyder@gmail.com, jianqian@hku.hk, chenyu.wei@virginia.edu

[1]Department of Computer Science, University of Virginia
[2]Department of AI and Data Science, The University of Hong Kong

Abstract

How can we train a new policy from a set of expert policies, when none of the experts fully solve the task and none are available at deployment? We propose algorithms to learn to stitch the best experts across states, and to further improve upon the stitched experts. We jointly train two policies: a *student*, which will ultimately be deployed, and a *selector*, which at each state selects an expert or the student to act in the environment. The selector is updated with standard reinforcement learning to maximize returns. The student is updated with standard reinforcement learning on its own actions, and with behavior cloning on actions taken by selected experts. Because the selector tends to select experts only if they outperform the student, the behavior cloning can automatically stop as the student outperforms the experts. Theoretically, we prove our algorithm competes with the best statewise stitching of the experts, and that the deployed student can improve beyond stitching under additional conditions. Empirically, we test on all combinations of four classic control environments and ten expert sets. Comparing algorithms by their per-environment-normalized returns, averaged over environments, our most robust algorithm matches or outperforms both max-aggregation (Cheng et al., 2020) and max$^+$-aggregation (Liu et al., 2024) on all ten expert sets.

1 Introduction

Learning a good policy from scratch can require many interactions with the environment. When expert policies are available, they can provide useful guidance and substantially accelerate learning, especially in the early stages, even if none of them fully solves the task. Moreover, in many settings there may be multiple imperfect experts with complementary strengths: one expert may perform well in some states, while another is better elsewhere. This raises a natural question: how can a learner best exploit multiple experts while still using reward feedback to improve beyond them? We study this question in a discounted MDP with K expert policies and one student. During training, the student can interact with the environment and query the experts for suggested actions; after training, however, the experts are unavailable, and only the student policy can be deployed.

Existing methods address different parts of this problem. Methods such as generalized policy improvement, max-value ensembles, and MaxFollowing combine expert policies through value estimates (Barreto et al., 2020; Hussing et al., 2024). In particular, MaxFollowing selects the expert with the largest estimated value at each state, which can exploit complementary experts and provide direct action supervision, but still keeps the resulting behavior tied to expert actions. Other methods, such as MaxAggregation and max$^+$-aggregation, use value supervision instead (Cheng et al., 2020; Liu et al., 2024): the learner searches over primitive actions using value targets derived from the experts. These approaches reveal a central tradeoff. Action supervision can make early learning fast by directly imitating useful experts, but by itself it cannot discover actions outside the experts.

Value supervision can move beyond the experts, but requires more exploration and may underuse useful expert actions early in training. Our goal is to combine these advantages: stitch the useful parts of multiple experts, transfer them into the student through action supervision, and let the student continue improving from reward feedback.

We propose Stitch-Imitate-Optimize (SIO), a method that combines expert stitching, action supervision, and reinforcement learning in a single training procedure. SIO trains a high-level selector over the K experts and the student. At each state, the selector chooses either an expert or the current student to act, and is updated by reinforcement learning to maximize return. The student is updated in two complementary ways: it applies reinforcement learning on the actions it takes itself, and behavior-clones the expert actions selected by the selector. Thus, expert supervision is adaptive rather than fixed: useful experts can be stitched and cloned early, weak experts can be avoided, and as the student improves, the selector can shift probability toward the student, reducing behavior cloning and turning the update increasingly into ordinary reinforcement learning.

Our theoretical analysis studies a population version of SIO. We first prove that the behavior policy induced by the selector competes with any policy representable as a statewise mixture of the experts, paying an additional residual term only for parts of the comparator outside the experts' convex hull. This formalizes the benefit of stitching complementary experts. We then give conditions under which the deployed student inherits the stitched behavior and can improve beyond the experts. Thus, the analysis separates two effects: learning to exploit the best statewise mixture of the experts, and transferring that behavior into a deployable student that can continue improving from reward feedback.

Empirically, we evaluate on four classic-control environments and ten expert-set types, including random experts, one strong expert mixed with random experts, and several complementary-expert constructions. Across these combinations, our methods are strongest when the experts contain useful but incomplete behavior: the selector can exploit complementary expertise, while the student can continue learning from reward. Using returns normalized per environment and averaged over environments, our most robust variants match or outperform both MaxAggregation and max$^+$-aggregation across the tested expert sets.

Our contributions are:

- We introduce SIO, which learns a return-trained selector over the K experts and the student. Unlike value-supervision methods such as MaxAggregation and max$^+$-aggregation, SIO directly turns selected expert actions into behavior-cloning targets. Unlike fixed BC+RL mixtures, the behavior-cloning signal is filtered by the selector and decreases statewise as the selector shifts probability toward the student.
- We analyze a population KL-regularized version of SIO. We prove that the selector-induced behavior policy has no regret to any statewise mixture of the experts, with an additional residual term only for comparator actions outside the experts' convex hull. We further give a deployment guarantee showing that, under deterministic experts, a value-gap condition, concentrability, and a decaying BC weight, the student can inherit the stitched behavior and improve beyond the experts.
- We evaluate SIO on all combinations of four classic-control environments and ten expert sets. By normalized return averaged over environments, SIO exceeds both MaxAggregation variants on every non-random expert set, while the robust variants AMSIO and ISSIO further improve robustness; ISSIO obtains the best mean rank across all environment–expert combinations.

2 Preliminaries

Notation. For a positive integer $N \in \mathbb{N}$, define $[N] = \{1, 2, \ldots, N\}$. We use Δ_X to denote the probability simplex over a set X, and write $\Delta_N = \Delta_{[N]}$.

Environment. We consider a discounted MDP with state space $\mathcal{S}$, action space $\mathcal{A}$, reward function $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, transition kernel $P : \mathcal{S} \times \mathcal{A} \to \Delta_{\mathcal{S}}$, initial state distribution $d_0 \in \Delta_{\mathcal{S}}$, and discount factor $\gamma \in [0, 1)$. A policy π maps each state $s \in \mathcal{S}$ to a distribution $\pi(\cdot|s) \in \Delta_{\mathcal{A}}$. When π is deterministic, we write $\pi(s) \in \mathcal{A}$ for the action selected at state s. For a policy π, define its state-value and state-action-value functions by $V^\pi(s) = \mathbb{E}^{\pi,P}\left[\sum_{i=1}^{\infty} \gamma^{i-1} R(s_i, a_i) \,\middle|\, s_1 = s\right]$, $Q^\pi(s, a) = \mathbb{E}^{\pi,P}\left[\sum_{i=1}^{\infty} \gamma^{i-1} R(s_i, a_i) \,\middle|\, (s_1, a_1) = (s, a)\right]$. The advantage function is $A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s)$. The value functions for the optimal policy are denoted as $Q^\star$ and $V^\star$. We use the normalized discounted occupancy measure $d^\pi_\varrho(s) = (1 - \gamma)\mathbb{E}^{\pi,P}\left[\sum_{i=1}^{\infty} \gamma^{i-1} \mathbf{1}[s_i = s] \,\middle|\, s_1 \sim \varrho\right]$, and write $d^\pi = d^\pi_{d_0}$. The normalized return is $J(\pi) = (1 - \gamma)\mathbb{E}_{s\sim d_0}[V^\pi(s)]$.

Experts. We are given K expert policies $\pi^{(1)}, \ldots, \pi^{(K)}$. During training, the student can query expert k at state s to obtain an action sample from $\pi^{(k)}(\cdot|s)$, but it does not observe the full distribution $\pi^{(k)}(\cdot|s)$. The experts are not available at deployment time; only the student policy is deployed.

We next review existing approaches for this setting.

2.1 MaxFollowing

MaxFollowing first evaluates the value function $V^{(k)} := V^{\pi^{(k)}}$ of each expert $k \in [K]$. It then follows the expert with the largest value at each state: let $k^\star(s) = \operatorname{argmax}_{k\in[K]} V^{(k)}(s)$ and set $\pi_{\mathrm{MF}}(s) = \pi^{(k^\star(s))}(s)$. This idea has been applied in autonomous driving (Li et al., 2018) and theoretically investigated in later work (Cheng et al., 2020; Hussing et al., 2024; Sengupta et al., 2025). When the evaluations of $V^{(k)}$ are accurate, MaxFollowing ensures $V^{\pi_{\mathrm{MF}}}(s) \geq V^{(k)}(s)$ for all s and $k \in [K]$.

A strength of MaxFollowing is that it provides direct *action supervision*: the student can directly imitate expert $k^\star(s)$'s action on state s. However, MaxFollowing requires estimating all expert value functions, and its policy is limited to following one of the experts at each state.

2.2 MaxAggregation

While MaxFollowing can uniformly improve over the experts, its policy is limited to following one of the experts at each state. This can be restrictive when the experts are poor. For example, when $K = 1$, MaxFollowing essentially reduces to imitating the only expert, and can perform poorly if that expert is poor. MaxAggregation (Cheng et al., 2020) is proposed to address this limitation.

Like MaxFollowing, MaxAggregation also estimates the value functions of all experts on each state. Instead of directly following the best expert, it performs one-step policy improvement over the maximum expert value. Let $f^{\max}(s) = \max_{k\in[K]} V^{(k)}(s)$. The target policy of MaxAggregation is

$$\pi_{\mathrm{MA}}(s) = \operatorname{argmax}_{a\in\mathcal{A}} A^{\max}(s, a) := \operatorname{argmax}_{a\in\mathcal{A}} R(s, a) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}[f^{\max}(s')] - f^{\max}(s).$$

With accurate value estimates, the resulting policy is no worse than any single expert. It gains additional benefits due to the one-step improvement. Unlike MaxFollowing, MaxAggregation uses *value supervision* rather than action supervision: The learner explores primitive actions (i.e., actions in the environment) and updates toward actions with larger $A^{\max}(s, a)$. This makes MaxAggregation more robust than directly imitating expert actions, but it requires more exploration from the student.

Cheng et al. (2020) use a PPO-style update with a GAE_λ estimator built from $A^{\max}$. The parameter λ controls the interpolation between bootstrapping from $f^{\max}$ when $\lambda \approx 0$ and using longer reward information from the student's own rollouts when $\lambda \approx 1$. Thus, λ balances RL and IL. This general version is called MAMBA in Cheng et al. (2020) while we call it MaxAggregation-NoSI later.

2.3 MaxAggregation with Self Inclusion (SI)

A limitation of MaxAggregation is that the bootstrap value $f^{\max}$ is static and does not move toward the optimal value $V^\star$ even when the student improves. This limits the ability of the student to

approach the true optimal policy. To address this, Liu et al. (2024) propose a *self-inclusive* version of MaxAggregation, which they call max$^+$-aggregation, while we will refer to it as MaxAggregation-SI later.

The key idea is to treat the current student as an additional, $(K+1)$-th expert. The aggregation value becomes $f_t^{\max,+}(s) = \max\left\{V^{(1)}(s), \ldots, V^{(K)}(s), V^{\pi_t}(s)\right\}$, where π_t is the student's current policy. As the student improves, its own value can dominate all fixed experts' values. In that regime, the update becomes closer to ordinary RL, allowing the learner to move beyond the fixed experts. MaxAggregation-SI remains a value-supervision method: it still relies on value estimation and primitive-action exploration, rather than directly cloning expert actions.

Action supervision versus value supervision. The above methods differ in how they use the experts. MaxFollowing uses action supervision, which reduces the need for exploration, but limits the student to the selected experts' actions. In contrast, MaxAggregation and MaxAggregation-SI use value supervision: the learner searches over primitive actions using a value target derived from the experts. This can improve beyond the experts, especially with self-inclusion, but it requires more exploration from the student and may underuse the experts during the early stages of training. Our approach combines the benefits of both supervisions, as we illustrate in the next section.

3 Proposed Algorithm: Stitch-Imitate-Optimize (SIO)

We improve prior methods by enabling the student to exploit the experts more effectively early in training, while preserving its ability to improve beyond them. To better exploit the experts, we propose to *stitch the experts* and use action supervision from the stitched expert policy to directly guide the student. To preserve the student's ability to improve beyond the experts, we incorporate *self-inclusion*, which lets the algorithm smoothly shift from imitation learning to reinforcement learning as the student improves. We elaborate these ideas below.

Stitching the experts. By *stitching*, we mean allowing different experts to act at different states, combining their strengths. MaxFollowing and MaxAggregation can somewhat *stitch* over experts in that they learn from the best expert value baseline $\max_k V^{(k)}(s)$ at every state s, and the maximizers differ among states. However, their theoretical guarantee only competes with the best single policy. We instead aim to compete with the *optimal stitching* of the experts, which could improve over MaxFollowing and MaxAggregation when the experts are *complementary*, i.e. good at different states. We achieve this stitching via a high-level *selector* that selects which expert should act at each state, with the goal of maximizing the return induced by this selection. Equivalently, this selector optimizes its policy in a meta-MDP where experts are actions. As the experts are not available at deployment, we further let the student behavior-clone (BC) the selected expert's action at each state, by maximizing the log-likelihood $\log \pi_\theta(a_i|s_i)$. We find this simple BC-based algorithm, **Stitch-Imitate-Optimize** with No Self-Inclusion (**SIO-NoSI**), empirically effective (Section 5.1).

Exploring beyond the experts. However, ideally we would like to explore beyond the expert actions. This theoretically allows the student to find the optimal policy of the environment rather than just the best stitching of the experts. Inspired by max$^+$-aggregation (Liu et al., 2024), we propose a self-inclusive (SI) version of SIO-NoSI, which we just call **SIO**. Algorithm 1 shows our practical implementation (we explain its q_ω later in this section). SIO modifies the BC variant of SIO-NoSI by: (1) adding an RL term for the student training, alongside the BC term; and (2) allowing the selector to select the student ($k_i = K+1$) as well, as opposed to only an expert ($k_i \leq K$). The selector is still trained to maximize returns from its selection. The student is trained to maximize return from its own actions (when it is selected by the selector), and to behavior-clone actions from experts (when an expert is selected by the selector). Concretely, SIO adds a PPO (Schulman et al., 2017) term, $\mathbf{1}[k_i = K+1] \min\{\rho_{\theta,i}\widetilde{A}_i,\ \rho_{\theta,i}^{\text{clip}}\widetilde{A}_i\}$, alongside the BC term, $\beta\mathbf{1}[k_i \leq K] \log \pi_\theta(a_i|s_i)$. We add β to balance the PPO and BC terms, though we use $\beta = 1$ except in our ablations. This is the student

Algorithm 1: SIO (Stitch-Imitate-Optimize).

1 **Inputs**: K experts $\pi^{(1)}, \ldots, \pi^{(K)}$.
2 **Hyperparameters**: learning rates α_{selector}, α_{student}, α_{critic}, BC weight β (we use $\beta = 1$),
3 GAE parameter λ, clipping factor ϵ_{clip}, batch size N, number of updates per batch M,
4 minibatch size B, student probability minimum ξ.
5 **Definitions**: $\text{clip}(x) := \max\{\min\{x, 1+\epsilon_{\text{clip}}\}, 1-\epsilon_{\text{clip}}\}$,
6 selector with minimum student probability ξ: $q_\omega(k \mid s) := (1-\xi)\, p_\omega(k \mid s) + \xi\, \mathbf{1}[k = K+1]$.
7 **Initialization**: selector network $p_\omega : \mathcal{S} \to \Delta_{K+1}$, student network $\pi_\theta : \mathcal{S} \to \Delta_{\mathcal{A}}$, critic $V_\phi : \mathcal{S} \to \mathbb{R}$.
8 **for** $t = 1, 2, \ldots$ **do**
9 `// data collection`
10 Observe state s_1.
11 **for** $i = 1, \ldots, N$ **do**
12 Sample expert or student $k_i \sim q_\omega(\cdot \mid s_i)$.
13 Sample action $a_i \sim \begin{cases} \pi^{(k_i)}(\cdot \mid s_i) & \text{if } k_i \le K, \\ \pi_\theta(\cdot \mid s_i) & \text{if } k_i = K+1. \end{cases}$
14 Observe reward and next state (r_i, s_i').
15 Draw s_{i+1} from initial distribution if s_i' is a terminal state; otherwise, $s_{i+1} = s_i'$.
16
17 `// data processing`
18 Compute advantage estimators: $(\hat{A}_i)_{i=1}^N = \text{GAE}_\lambda((s_i, r_i, s_i')_{i=1}^N, V_\phi)$.
19 Define $\hat{V}_i = V_\phi(s_i) + \hat{A}_i$ for $i \in [N]$.
20 Define $q_{\text{old}} = q_\omega$ and $\pi_{\text{old}} = \pi_\theta$.
21
22 **for** $j = 1, \ldots, M$ **do**
23 Sample a minibatch $\mathcal{B} \subset [N]$ with size $|\mathcal{B}| = B$.
24 Define $\rho_{\omega,i} = \frac{q_\omega(k_i|s_i)}{q_{\text{old}}(k_i|s_i)}$, $\rho_{\omega,i}^{\text{clip}} = \text{clip}(\rho_{\omega,i})$, $\rho_{\theta,i} = \frac{\pi_\theta(a_i|s_i)}{\pi_{\text{old}}(a_i|s_i)}$, $\rho_{\theta,i}^{\text{clip}} = \text{clip}(\rho_{\theta,i})$ for $i \in \mathcal{B}$.
25 Compute minibatch-normalized advantage $\widetilde{A}_i$ from $\hat{A}_i$.
26 `// selector network update`
27 $\omega \leftarrow \omega + \alpha_{\text{selector}} \nabla_\omega \frac{1}{B} \sum_{i \in \mathcal{B}} \min\left\{\rho_{\omega,i}\widetilde{A}_i, \rho_{\omega,i}^{\text{clip}}\widetilde{A}_i\right\}$.
28 `// student network update`
29 $\theta \leftarrow \theta + \alpha_{\text{student}} \nabla_\theta \frac{1}{B} \sum_{i \in \mathcal{B}} \left(\mathbf{1}[k_i = K+1] \min\left\{\rho_{\theta,i}\widetilde{A}_i, \rho_{\theta,i}^{\text{clip}}\widetilde{A}_i\right\} + \beta \mathbf{1}[k_i \le K] \log \pi_\theta(a_i|s_i)\right)$.
30 `// critic network update`
31 $\phi \leftarrow \phi - \alpha_{\text{critic}} \nabla_\phi \frac{1}{B} \sum_{i \in \mathcal{B}} \left(V_\phi(s_i) - \hat{V}_i\right)^2$.
32 **Deployment:** π_θ.

update at Line 29. The student is deployed at test time. We find SIO both theoretically (Section 4) and empirically (Section 5) effective.

Intuition. Unlike approaches that pretrain with BC and then post-train with RL, and unlike approaches that naively train with BC and RL together, SIO can automatically use BC only when it is increasing returns: this can vary over states, experts, and training steps. A simplified view is two phases. Phase one: the student is worse than the experts, so the selector mostly stitches expert actions, and the student clones. Phase two: once the selector has learned to stitch the experts and the student has cloned this behavior, the student will be selected with higher probability, $p_\omega(K+1|s)$, due to its improved performance. Every time the student is selected, it trains with RL, so can eventually surpass the experts, which would cause the selector to further reduce the weight on the experts and hence the

probability of behavior-cloning. However, to emphasize, the algorithm might do even better than a simple two-phase approach over all states, experts, and training steps.

Prioritizing the deployment policy. The naive implementation of SIO initializes its selector policy with similar probabilities over all of the experts, including the student policy as if it were an expert. This is likely often undesirable because we ultimately will deploy the student policy. Even though the naive implementation of SIO can automatically learn to favor the student once the student has surpassed the experts (by its RL term, or by learning to stitch the best experts through BC, or both), this mechanism alone might not prioritize the student enough. To handle this, we add a student probability minimum hyperparameter, ξ, which we find empirically increases robustness to bad experts, especially if there are many of them (Section 5.1). This hyperparameter is analogous to λ_{MA} in MaxAggregation, which balances RL and IL. Specifically, when selecting an expert k_i to act in the environment, instead of using the original selector $p_\omega(\cdot \mid s)$, we select

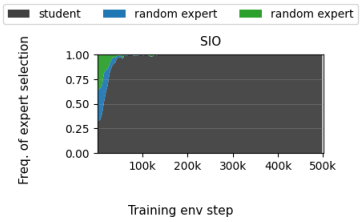

Figure 1: SIO quickly avoids BC given random experts. `CartPole-v1` with $\xi = 0.0$.

$$k_i \sim q_\omega(k \mid s_i) \;=\; (1-\xi)\, p_\omega(k \mid s_i) \;+\; \xi\, \mathbf{1}[k = K+1]\,,$$

so the student policy is selected with probability at least ξ (Line 6, Line 12).

4 Theoretical Analysis

In this section, we analyze a population version of Algorithm 1, shown in Algorithm 2. The goal is to formalize the two mechanisms behind SIO. First, the high-level selector policy learns to stitch the best experts across states. Second, the low-level student policy learns from this stitched behavior policy while also improving beyond the experts through self-inclusion.

The population version Algorithm 2 is a *full-information* idealization of Algorithm 1. The behavior policy μ_t defined in Line 7 corresponds to the policy used to collect data in Algorithm 1. The population version assumes access to the exact advantage function $A^{\mu_t}(s,a)$, and assumes that the selector-policy update Line 4 and the student-policy update Line 5 are solved exactly, without sampling error, optimization error, or function-approximation error. Additionally, it replaces PPO clipping with KL regularization, which lets us view the updates as mirror-ascent steps. Lastly, the BC weight $\beta_t(s)$ is made both time and state dependent, as needed by our theoretical analysis.

The regret of the behavior policy We first show that the behavior policy μ_t can compete with any policy μ that can be represented as a statewise mixture of the experts. If μ cannot be represented this way, the regret pays only for the residual part of μ that is not covered by the experts.

Theorem 1. *For any policy $\mu : \mathcal{S} \to \Delta_{\mathcal{A}}$, assume that $\mu(\cdot|s)$ can be decomposed as*

$$\mu(\cdot|s) = \textstyle\sum_{k=1}^{K} p_\mu(k|s)\pi^{(k)}(\cdot|s) + p_\mu(K+1|s)\mu_\perp(\cdot|s) \tag{1}$$

for some $p_\mu(\cdot|s) \in \Delta_{K+1}$ and $\mu_\perp(\cdot|s) \in \Delta_{\mathcal{A}}$. Then the update in Line 4 with $p_1(\cdot|s) = \frac{\mathbf{1}}{K+1}$ ensures

$$\textstyle\sum_{t=1}^{T} \left(J(\mu) - J(\mu_t)\right) \le 2L^2 T\eta + \eta^{-1}\log(K+1) + \mathbb{E}_{s\sim d^\mu}\left[p_\mu(K+1|s)\mathrm{Reg}_{\mathrm{stu}}(\mu_\perp; s)\right], \tag{2}$$

where $\mathrm{Reg}_{\mathrm{stu}}(\pi; s) := \sum_{t=1}^{T} \langle \pi(\cdot|s) - \pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle$ and $L = \max_{\pi,s,a} |A^\pi(s,a)|$.

Theorem 1 formalizes the statewise stitching effect of the selector policy. The first two terms in (2), $2L^2T\eta + \eta^{-1}\log(K+1)$, are the standard mirror-ascent regret for learning among $K+1$ experts. The last term in (2) is the price of comparing against the residual of μ that is not represented by the

Algorithm 2: Population version of SIO for theoretical analysis

1 **Hyperparameters**: KL-regularization parameter η, behavior-cloning weights $\beta_t(s)$.
2 **Initialization**: $p_1(\cdot|s)$ and $\pi_1(\cdot|s)$ for all states s.
3 **for** $t = 1, 2, \ldots, T$ **do**
4 $\quad p_{t+1} = \operatorname*{argmax}_p \mathbb{E}_{s\sim d^{\mu_t}} \left[\mathbb{E}_{k\sim p(\cdot|s)} \left[A_t^{\mu_t}(s,k)\right] - \eta^{-1} D_{\mathrm{KL}}(p(\cdot|s) \parallel p_t(\cdot|s))\right],$
5 $\quad \pi_{t+1} = \operatorname*{argmax}_\pi \mathbb{E}_{s\sim d^{\mu_t}} \Big[p_t(K+1|s) \left(\mathbb{E}_{a\sim\pi(\cdot|s)} \left[A^{\mu_t}(s,a)\right] - \eta^{-1} D_{\mathrm{KL}}(\pi(\cdot|s) \parallel \pi_t(\cdot|s))\right)$
$\qquad\qquad + \beta_t(s) \sum_{k=1}^K p_t(k|s) \mathbb{E}_{a\sim\pi^{(k)}(\cdot|s)}[\log \pi(a|s)]\Big]$
6 $\quad$ where μ_t is the policy induced by the expert distribution p_t and student policy π_t:
7 $\quad \mu_t(a|s) = \sum_{k=1}^K p_t(k|s)\pi^{(k)}(a|s) + p_t(K+1|s)\pi_t(a|s)$
8 $\quad$ and $A_t^{\mu_t}$ is the expert-level advantage function:
9 $\quad A_t^{\mu_t}(s,k) = \begin{cases} \langle \pi^{(k)}(\cdot|s), A^{\mu_t}(s,\cdot)\rangle & \text{if } k \in [K], \\ \langle \pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle & \text{if } k = K+1. \end{cases}$

experts combination. By the decomposition (1), if $\mu(\cdot|s)$ lies in the convex hull of $\{\pi^{(k)}(\cdot|s)\}_{k=1}^K$, then we can choose $p_\mu(K+1|s) = 0$ and the residual cost at state s vanishes.

Note that this guarantee depends only on the selector-policy update Line 4, not on the student-policy update Line 5. Intuitively, if the student policy is still poor, the selector can place little mass on the student branch and focus on stitching among the fixed experts. Thus, the behavior policy can already compete with the best expert stitching before the student becomes useful.

The regret of the student policy Now we turn to the student policy π_t. We show that, under certain assumptions, the student can inherit the stitched behavior and eventually improve beyond the experts.

Theorem 2. *Let $\mu^\star$ be the optimal policy. Assume 1) a concentrability coefficient C that bounds the state-density ratio between any two policies, 2) minimum value gap $V^\star(s) - Q^\star(s,a) \geq \Delta_{\min}$ for any state s and any suboptimal action a, 3) deterministic experts policies, 4) $\beta_t(s) = p_t(K+1|s)t^{-\zeta}$ for some $\zeta \in (0, \frac{1}{2}]$. Then the regret of the student policy satisfies*

$$\sum_{t=1}^T \left(J(\mu^\star) - J(\pi_t)\right) \leq O\left(\sqrt{\frac{C^4 L^3}{\Delta_{\min}} \left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T)\right) T^{1+\zeta}}\right),$$

where $B_{\exp}(T) = L^2 T\eta + \eta^{-1}\log(K|\mathcal{A}|)$, $B_{\mathrm{opt}}(T) = C\Delta_{\min}^{-1} B_{\exp}(T) + (C\Delta_{\min}^{-1})^{1+1/\zeta}$ and $\sigma = \mathbf{1}\left[\exists s, \mu^\star(s) \notin \{\pi^{(k)}(s)\}_{k=1}^K\right]$.

The proof is deferred to Section D.2. The regret bound has two components, with $B_{\exp}(T) \leq B_{\mathrm{opt}}(T)$. The smaller term $B_{\exp}(T)$ corresponds to the easier "stitching over experts" regime, and the larger term $B_{\mathrm{opt}}(T)$ corresponds to the harder "exploring beyond experts" regime. Since the harder regime is only necessary when the optimal action $\mu^\star(s)$ does not lie in the expert choice $\{\pi^{(k)}(s)\}_{k=1}^K$, the contribution of $B_{\mathrm{opt}}(T)$ is scaled by σ.

A key in the proof is to show that $\sum_t (1 - p_t(K+1|s)) \|\overline{\mu}_t(\cdot|s) - \pi_t(\cdot|s)\|^2 = o(T)$, where $\overline{\mu}_t(\cdot|s)$ is the weighted average over expert policies with weights $\propto p_t(k|s)$ (Lemma 6). This summation being $o(T)$ indicates that either $1 - p_t(K+1|s)$ or $\|\overline{\mu}_t(\cdot|s) - \pi_t(\cdot|s)\|^2$ is close to zero in the long run. The former indicates that the student policy is dominating all experts, and the latter indicates that the student policy is close to the experts' mixture policy. They cover both cases where the experts choices *do not* and *do* include the optimal action, respectively.

Purpose and limitation of the theoretical analysis Our analysis focuses on the population-level *dynamics* of the selector and the student. The goal is to verify that their coupled updates can indeed converge to desired solutions. The analysis does not address *sample complexity* though, as we assume

access to full-information advantage functions and exact policy updates. We leave a full-stack analysis for future work.

5 Experiments

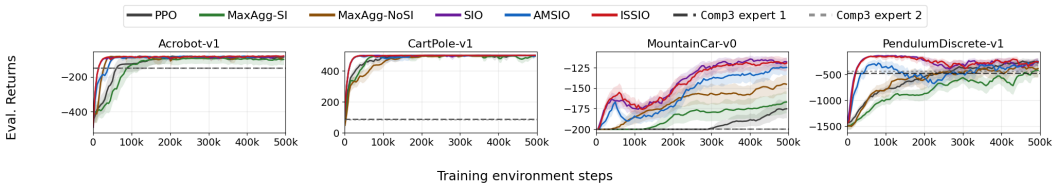

Figure 2: `Comp3` learning curves, with all learning rates at 10^{-3}. MaxAgg $\lambda_{\mathrm{MA}} = 0.95$, and SIO $\xi = 0$. All algorithms learn quickly on `Acrobot-v1` and `CartPole-v1`, though our algorithms are all among the fastest. On `MountainCar-v0` and `PendulumDiscrete-v1`, our algorithms learn faster than both MaxAggregation (regardless of SI) and PPO. Thirty seeds, 95% bootstrap CIs.

We test on four discrete-action, classic control tasks (Brockman et al., 2016; Towers et al., 2024): `CartPole-v1`, `Acrobot-v1`, `MountainCar-v0`, and a discrete-action version of `Pendulum-v1`. We consider the following expert sets:

1. `R`: A single random expert: a policy that acts uniformly randomly.
2. `2R`: Two random experts.
3. `8R`: Eight random experts.
4. `G`: A single great expert: a DQN net trained on the environment, which had its learning rate tuned to maximize the expert's returns.
5. `G2R`: The great expert and two random experts.
6. `G7R`: The great expert and seven random experts.
7. `Comp1`: the great expert modified into two new, complementary experts. One position value of the state space is split into two halves, and each of the two experts acts uniformly randomly on opposite halves (remaining great on its other half). For example, for `CartPole-v1`, the split is the pole angled left vs. right.
8. `Comp2`: the same as `Comp1`, but splitting on the velocity value of the state space. For example, for `CartPole-v1`, the split is the pole moving left vs. right.
9. `Comp3`: roughly the same as `Comp1`, but a checkerboard pattern over the state space instead of a left-right split. For example, for `CartPole-v1`, each expert is great on only half of twenty bins over cart position and twenty bins over pole angle.
10. `Comp4` (complementary experts 4): roughly the same as `Comp3`, but eight experts instead of two, where each expert is great on only one-eighth of the checkerboard.

To measure learning speed, we compare average evaluation returns during all of training. In our experiments, on all non-random expert sets, our basic algorithm SIO with $\xi = 0.5$ attains higher means than MaxAggregation-SI (self-inclusion) and MaxAggregation-NoSI (no self-inclusion), with only one overlapping CI (Table 1). We also include two additional variants, Advantage-Masked SIO (**AMSIO**) and Importance-Weight SIO (**ISSIO**), that differ from SIO only in the BC term of the student-actor update (Line 29): they replace $\beta \mathbf{1}[k_i \le K] \log \pi_\theta(a_i \mid s_i)$ with (new terms in orange):

$$\underbrace{\mathbf{1}\left[\widetilde{A}_i > 0\right] \beta \mathbf{1}[k_i \le K] \log \pi_\theta(a_i \mid s_i)}_{\text{AMSIO}} \quad \text{or} \quad \underbrace{\beta \mathbf{1}[k_i \le K] \min\{\rho_{\theta,i},\ 1 + \epsilon_{\text{clip}}\}}_{\text{ISSIO}}.$$

We find both even more robust than SIO. All but one of AMSIO's mean returns in Table 1 are higher than both MaxAggregation-SI and MaxAggregation-NoSI. Further, on that exception, `R`, AMSIO's returns are smaller by less than 1% (and the CIs overlap). Out of all of our algorithms and baselines, ISSIO has the best mean rank over all environment–expert combinations (Table 4). With Adam's

Table 1: For each environment, evaluation returns are averaged over all of training and over the learning-rate grid (Section E.1), then normalized so a uniformly random policy's returns are 0, and DQN's returns are 100 under the same averaging. The per-environment normalized returns are then averaged over environments. Plain PPO under the same procedure gets 49.5 normalized returns. All tables and figures use means and 95% bootstrap CIs over ten seeds unless otherwise specified. For concerns about outlier environments, and for default learning rate (10^{-3}) results, see Section C.

Experts	MaxAgg-SI $\lambda_{\text{MA}} = 0.95$	MaxAgg-NoSI $\lambda_{\text{MA}} = 0.95$	SIO $\xi = 0.5$	AMSIO $\xi = 0.5$	ISSIO $\xi = 0.5$
R	45.5 [43.2, 47.7]	**55.1 [53.6, 56.7]**	38.1 [36.1, 40.0]	54.8 [52.7, 56.9]	46.7 [44.4, 49.2]
2R	42.7 [40.6, 44.8]	52.5 [50.5, 54.3]	36.3 [34.0, 38.6]	**53.2 [50.9, 55.6]**	43.4 [40.9, 45.9]
8R	42.2 [40.0, 44.3]	48.9 [46.9, 51.0]	33.3 [31.3, 35.5]	**50.5 [48.7, 52.3]**	38.6 [36.7, 40.6]
G	52.7 [49.7, 55.7]	76.9 [74.7, 79.0]	**134.3 [129.7, 139.0]**	104.1 [101.7, 106.5]	130.9 [125.9, 135.7]
G2R	52.5 [49.8, 55.3]	73.5 [71.9, 75.1]	96.7 [93.4, 100.1]	87.4 [84.6, 90.4]	**100.5 [97.5, 103.6]**
G7R	51.2 [48.6, 53.8]	64.9 [63.1, 66.7]	72.5 [69.7, 75.4]	72.8 [70.7, 75.0]	**76.9 [74.2, 79.7]**
Comp1	50.0 [47.5, 52.6]	73.7 [70.7, 76.7]	96.0 [94.5, 97.5]	96.0 [94.1, 97.8]	**97.9 [96.3, 99.7]**
Comp2	49.2 [46.8, 51.4]	68.8 [66.3, 71.5]	99.9 [98.0, 102.0]	87.8 [85.7, 90.0]	**100.2 [98.4, 102.2]**
Comp3	50.8 [48.3, 53.2]	81.0 [78.1, 83.7]	104.7 [102.4, 106.9]	102.1 [100.0, 104.3]	**111.9 [108.5, 115.2]**
Comp4	44.1 [41.9, 46.3]	58.9 [57.3, 60.5]	62.4 [60.4, 64.6]	**68.8 [67.0, 70.8]**	67.9 [66.0, 70.0]

(Kingma & Ba, 2014) default learning rate for all algorithms, SIO, AMSIO, and ISSIO all beat both MaxAggregation-SI and MaxAggregation-NoSI in mean returns on more than 80% of the environment–expert combinations (Table 7). As expected, we also find that the better the experts, the better MaxAggregation-NoSI performs compared to MaxAggregation-SI.

For our main results on all algorithms, we sweep the actor and critic learning rates over a fixed grid, tying all actor learning rates into one, and all critic learning rates into one. We show the per-learning-rate results[1] in Section F, and the frequency of each expert or student being selected in Section G.

5.1 Additional Experiments

We ablate the BC coefficient and self-inclusion in Table 2, and the student-selection floor in Table 3.

Table 2: SIO ablations ($\xi = 0$). Again normalized by DQN (100) and a random policy (0). *Left*: our default BC coefficient $\beta = 1$ vs. smaller coefficients. *Right*: our default self-inclusive (SI) vs. NoSI.

Experts	$\beta = 1$	$\beta = 0$	$\beta = 0.5$	SI	NoSI
R	34.0 [31.7, 36.4]	**52.0 [49.4, 54.4]**	38.1 [35.7, 40.6]	**34.0 [31.7, 36.4]**	-10.3 [-11.5, -9.1]
2R	26.2 [23.8, 28.5]	**44.7 [41.9, 47.5]**	29.2 [26.7, 31.8]	**26.2 [23.8, 28.5]**	-11.8 [-12.7, -10.8]
8R	9.7 [7.4, 12.1]	**27.5 [25.1, 29.9]**	13.6 [11.3, 15.8]	**9.7 [7.4, 12.1]**	-9.5 [-10.8, -8.1]
G	**141.5 [137.5, 145.5]**	29.5 [27.4, 31.7]	133.1 [129.7, 136.5]	141.5 [137.5, 145.5]	**160.8 [159.9, 161.6]**
G2R	**106.7 [103.0, 110.6]**	33.4 [31.4, 35.5]	98.2 [94.9, 101.8]	106.7 [103.0, 110.6]	**107.1 [103.4, 111.2]**
G7R	77.7 [74.6, 81.1]	27.5 [24.6, 30.3]	**80.5 [76.6, 84.4]**	77.7 [74.6, 81.1]	**81.6 [77.7, 85.7]**
Comp1	**111.4 [108.3, 114.8]**	44.1 [41.3, 46.8]	106.0 [102.6, 109.4]	111.4 [108.3, 114.8]	**116.6 [113.1, 120.4]**
Comp2	**107.2 [104.7, 109.7]**	34.2 [31.5, 36.7]	102.2 [100.3, 104.1]	107.2 [104.7, 109.7]	**114.9 [112.5, 117.4]**
Comp3	**133.2 [129.0, 137.4]**	47.8 [45.6, 50.1]	128.0 [124.5, 131.4]	133.2 [129.0, 137.4]	**158.3 [157.5, 159.1]**
Comp4	**55.9 [54.2, 57.8]**	39.0 [36.8, 41.2]	53.1 [51.7, 54.5]	**55.9 [54.2, 57.8]**	54.8 [53.6, 55.9]

[1]With their best learning rate settings, algorithms that use the experts occasionally learn faster than PPO even when the experts are all random. We speculate this is because the random experts cause more exploration. Note that, for simplicity, we use no entropy coefficient in our algorithms nor our baselines.

Varying the BC coefficient: $\beta \neq 1$. All of our main results (e.g. Table 1) use $\beta = 1$, which works well, perhaps because SIO automatically tends to avoid training data from bad experts. However, the left side of Table 2 shows that, depending on the expert set, SIO can learn even better with a tuned β.

No self-inclusion (NoSI). The right side of Table 2 compares self-inclusion (SI) to NoSI. As expected, NoSI seems to work better when the highest-performing individual expert is stronger, and also when each of the experts is individually stronger. The negative values for some NoSI cells are NoSI performing worse than a uniformly random policy, which is common for many algorithms and learning rates on `Pendulum-v1`, where often the agent must counterintuitively rotate away from the goal angle to gain momentum.

No probability minimum for selecting the student, $\xi \neq 0.5$. Table 3 demonstrates the benefits and downsides of the ξ hyperparameter: the student probability minimum helps on the random-expert sets, while $\xi = 0$ is better for most great, mixed, and complementary expert sets.

Table 3: SIO with a student-selection floor ($\xi = 0.5$) vs. no floor ($\xi = 0$).

Experts	$\xi = 0$	$\xi = 0.5$
`R`	34.0 [31.7, 36.4]	**38.1 [36.1, 40.0]**
`2R`	26.2 [23.8, 28.5]	**36.3 [34.0, 38.6]**
`8R`	9.7 [7.4, 12.1]	**33.3 [31.3, 35.5]**
`G`	**141.5 [137.5, 145.5]**	134.3 [129.7, 139.0]
`G2R`	**106.7 [103.0, 110.6]**	96.7 [93.4, 100.1]
`G7R`	**77.7 [74.6, 81.1]**	72.5 [69.7, 75.4]
`Comp1`	**111.4 [108.3, 114.8]**	96.0 [94.5, 97.5]
`Comp2`	**107.2 [104.7, 109.7]**	99.9 [98.0, 102.0]
`Comp3`	**133.2 [129.0, 137.4]**	104.7 [102.4, 106.9]
`Comp4`	55.9 [54.2, 57.8]	**62.4 [60.4, 64.6]**

6 Discussion and limitations

Limitations. We assume expert actions are observable at training time, though this is a common setting (for example, with large language models). While we test great experts, random experts, and expert sets that combine them in various ways, future work should perhaps also test experts that are only *moderately* good at all states. Future work should perhaps also test complementary expert sets that require exploration significantly beyond the experts for successful stitching. Additionally, we test only synthetic experts, and only on low-dimensional environments with discrete actions.

Future algorithms. Combining the advantage-masking of AMSIO with the importance-sampling BC term of ISSIO might improve over both algorithms. Alternatively, using off-policy learning for the student might more efficiently use expert data than our initial implementation. TD3+BC (Fujimoto & Gu, 2021) is framed as an offline RL algorithm, but might be a natural fit for the student given how beneficial we find our BC term. Uncertainty-based approaches (Jin et al., 2021) for the student similarly might be important.

Acknowledgments

Part of this research was performed while BS was visiting the Institute for Mathematical and Statistical Innovation (IMSI), which is supported by the National Science Foundation (Grant No. DMS-2425650).

We thank RLBRew reviewers for their helpful comments.

References

André Barreto, Shaobo Hou, Diana Borsa, David Silver, and Doina Precup. Fast reinforcement learning with generalized policy updates. *Proceedings of the National Academy of Sciences*, 117 (48):30079–30087, 2020.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym. *arXiv*, June 2016. DOI: 10.48550/arXiv.1606.01540.

Jacob Buckman, Carles Gelada, and Marc G Bellemare. The importance of pessimism in fixed-dataset policy optimization. In *International Conference on Learning Representations*, 2021.

C Cheng, X Yan, N Wagener, and B Boots. Fast policy learning through imitation and reinforcement. In *Uncertainty in artificial intelligence*, 2019.

Ching-An Cheng, Andrey Kolobov, and Alekh Agarwal. Policy improvement via imitation of multiple oracles. *Advances in Neural Information Processing Systems*, 33:5587–5598, 2020.

CleanRL contributors. CleanRL DQN implementation, 2026a. URL https://github.com/vwxyzjn/cleanrl/blob/fe8d8a03c41a7ef5b523e2e354bd01c363e786bb/cleanrl/dqn.py. GitHub commit fe8d8a03c41a7ef5b523e2e354bd01c363e786bb.

CleanRL contributors. CleanRL PPO implementation, 2026b. URL https://github.com/vwxyzjn/cleanrl/blob/fe8d8a03c41a7ef5b523e2e354bd01c363e786bb/cleanrl/ppo.py. GitHub commit fe8d8a03c41a7ef5b523e2e354bd01c363e786bb.

Lakshita Dodeja, Ondrej Biza, Shivam Vats, Stephen Hart, Stefanie Tellex, Robin Walters, Karl Schmeckpeper, and Thomas Weng. When life gives you BC, make Q-functions: Extracting Q-values from behavior cloning for On-Robot reinforcement learning. *arXiv preprint arXiv:2605.05172*, 2026. DOI: 10.48550/arXiv.2605.05172.

Scott Fujimoto and Shixiang (Shane) Gu. A Minimalist Approach to Offline Reinforcement Learning. *Advances in Neural Information Processing Systems*, 34:20132–20145, December 2021. URL https://proceedings.neurips.cc/paper/2021/hash/a8166da05c5a094f7dc03724b41886e5-Abstract.html.

Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL http://jmlr.org/papers/v23/21-1342.html.

Marcel Hussing, Michael Kearns, Aaron Roth, Sikata Bela Sengupta, and Jessica Sorrell. Oracle-Efficient Reinforcement Learning for Max Value Ensembles. *ArXiv*, abs/2405.16739, 2024. URL https://api.semanticscholar.org/CorpusID:270062734.

Ying Jin, Zhuoran Yang, and Zhaoran Wang. Is Pessimism Provably Efficient for Offline RL? In *International Conference on Machine Learning*, pp. 5084–5096. PMLR, July 2021. URL https://proceedings.mlr.press/v139/jin21e.html.

Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv*, December 2014. DOI: 10.48550/arXiv.1412.6980.

Andrey Kurenkov, Ajay Mandlekar, Roberto Martin-Martin, Silvio Savarese, and Animesh Garg. AC-Teach: A bayesian actor-critic method for policy learning with an ensemble of suboptimal teachers. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 717–734. PMLR, 2020. URL https://proceedings.mlr.press/v100/kurenkov20a.html.

Guohao Li, Matthias Mueller, Vincent Casser, Neil Smith, Dominik L Michels, and Bernard Ghanem. Oil: Observational imitation learning. *arXiv preprint arXiv:1803.01129*, 2018.

Xuefeng Liu, Takuma Yoneda, Chaoqi Wang, Matthew Walter, and Yuxin Chen. Active policy improvement from multiple black-box oracles. In *International Conference on Machine Learning*, pp. 22320–22337. PMLR, 2023.

Xuefeng Liu, Takuma Yoneda, Rick Stevens, Matthew Walter, and Yuxin Chen. Blending imitation and reinforcement learning for robust policy improvement. In *The Twelfth International Conference on Learning Representations*, 2024.

Chiara Mignacco. *A mathematical study of policy orchestration for reinforcement learning*. PhD thesis, Université Paris-Saclay, Paris, France, December 2025. URL https://theses.hal.science/tel-05467153.

Simon Schmitt, Jonathan J. Hudson, Augustin Zidek, Simon Osindero, Carl Doersch, Wojciech M. Czarnecki, Joel Z. Leibo, Heinrich Kuttler, Andrew Zisserman, Karen Simonyan, and S. M. Ali Eslami. Kickstarting Deep Reinforcement Learning. *arXiv*, March 2018. DOI: 10.48550/arXiv.1803.03835.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal Policy Optimization Algorithms. *arXiv*, July 2017. DOI: 10.48550/arXiv.1707.06347.

Sikata Bela Sengupta, Zakaria Mhammedi, and Teodor Vanislavov Marinov. Oracle-Efficient Adversarial Reinforcement Learning via Max-Following. In *The Exploration in AI Today Workshop at ICML 2025*, 2025. URL https://openreview.net/forum?id=KdYtu1HiE9.

Idan Shenfeld, Zhang-Wei Hong, Aviv Tamar, and Pulkit Agrawal. Tgrl: An algorithm for teacher guided reinforcement learning. In *International Conference on Machine Learning*, pp. 31077–31093. PMLR, 2023.

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U. Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun Kg, Rodrigo Perez-Vicente, Andrea Pierré, Sander Schulhoff, Jun Jet Tai, Hannah Tan, and Omar G. Younis. Gymnasium: A Standard Interface for Reinforcement Learning Environments. *arXiv*, July 2024. DOI: 10.48550/arXiv.2407.17032.

Supplementary Materials

The following content was not necessarily subject to peer review.

A Full MaxAggregation Pseudocode

Algorithm 3: MaxAggregation (SI-only lines in **red**)

Inputs: K experts $\pi^{(1)}, \ldots, \pi^{(K)}$.
Hyperparameters: learning rates α_{actor}, α_{experts}, α_{critic} (SI only), PPO clipping factor ϵ_{clip}, batch size N, number of updates per batch M, minibatch size B, pretrain episodes per expert N_{pretrain}, GAE parameter λ_{MA}.
Initialization: expert critic $V_\psi : \mathcal{S} \to \mathbb{R}^K$ with heads $V_\psi^{(k)}$ estimating $V^{(k)}$ for $k \in [K]$, student $\pi_\theta : \mathcal{S} \to \Delta_{\mathcal{A}}$, empty per-expert suffix replay buffers $D_1, \ldots, D_K$ (each capped at N transitions, dropping oldest whole suffixes); $t \leftarrow 0$, cycling expert index $i \leftarrow 0$, next PPO main-env step $t_{\text{next}} \leftarrow 0$.
Initialization (SI only): student-critic $V_\phi : \mathcal{S} \to \mathbb{R}$.
`// pretrain`
for $k = 1, \ldots, K$ **do**
  Run N_{pretrain} full episodes under expert $\pi^{(k)}$, appending the resulting transitions to D_k.
  Update $V_\psi^{(k)}$ with λ_{MA}-returns on D_k.
while *True* **do**
  `// episode rollout (RIRO)`
  Pick expert $k \leftarrow 1 + (i \bmod K)$; $i \leftarrow i + 1$.
  Sample RIRO horizon split point $h \in \mathsf{Uniform}(0, H)$; reset env; $\tau \leftarrow$ empty suffix buffer.
  for $j = 0, 1, 2, \ldots$ *until episode end* **do**
    if $j < h$ **then**
      `// roll-in: student acts; transition not pushed to replay`
      $a_j \sim \pi_\theta(\cdot \mid s_j)$.
    else
      `// roll-out: fixed expert k`
      $a_j \sim \pi^{(k)}(\cdot \mid s_j)$.
    Step env, observing (r_j, s_{j+1}); $t \leftarrow t + 1$.
    if $j \geq h$ **then**
      append observed transition to τ.
    if $t \geq t_{\text{next}}$ **then**
      Run PPOPHASE; $t_{\text{next}} \leftarrow t + N$.
  `// critic update`
  Append τ to D_k; drop oldest whole suffixes until D_k contains at most N transitions.
  For each suffix in D_k, update $V_\psi^{(k)}$ with λ_{MA}-returns.

Procedure PPOPHASE**:**
  `// data collection`
  Collect $(s_i, a_i, r_i, s_i')_{i=1}^N$ under the student π_θ.
  `// data processing`
  Define $f^{\max}(s) = \max_{k \in [K]} V_\psi^{(k)}(s)$ and $f_t^{\max,+}(s) = \max\big(f^{\max}(s),\, V_\phi(s)\big)$.
  Compute advantage estimators: $(\hat{A}_i)_{i=1}^N = \text{GAE}_{\lambda_{\text{MA}}}((s_i, r_i, s_i')_{i=1}^N, f^{\max})$ (non-SI) or $\text{GAE}_{\lambda_{\text{MA}}}((s_i, r_i, s_i')_{i=1}^N, f_t^{\max,+})$ (SI).
  Also compute $(\hat{A}_i^\phi)_{i=1}^N = \text{GAE}_{\lambda_{\text{MA}}}((s_i, r_i, s_i')_{i=1}^N, V_\phi)$, and define $\hat{V}_i = V_\phi(s_i) + \hat{A}_i^\phi$ for $i \in [N]$.
  Define $\pi_{\text{old}} = \pi_\theta$.
  for $j = 1, \ldots, M$ **do**
    Sample a minibatch $\mathcal{B} \subset [N]$ with size $|\mathcal{B}| = B$.
    Define $\rho_{\theta,i} = \frac{\pi_\theta(a_i|s_i)}{\pi_{\text{old}}(a_i|s_i)}$, $\rho_{\theta,i}^{\text{clip}} = \text{clip}(\rho_{\theta,i})$ for $i \in \mathcal{B}$.
    Compute minibatch-normalized advantage $\widetilde{A}_i$ from $\hat{A}_i$.
    `// student network update`
    $\theta \leftarrow \theta + \alpha_{\text{actor}} \nabla_\theta \frac{1}{B} \sum_{i \in \mathcal{B}} \min\big\{\rho_{\theta,i} \widetilde{A}_i, \rho_{\theta,i}^{\text{clip}} \widetilde{A}_i\big\}$.
    `// student-critic update`
    $\phi \leftarrow \phi - \alpha_{\text{critic}} \nabla_\phi \frac{1}{B} \sum_{i \in \mathcal{B}} \big(V_\phi(s_i) - \hat{V}_i\big)^2$.

Deployment: π_θ.

For infinite-horizon tasks, we follow MAMBA and RPI in tracking a moving average of the student episode lengths to periodically estimate the empirical horizon lengths, then sample the RIRO splitting point geometrically from that length (rather than drawing uniformly from $[0, H]$).

B Related Work

B.1 Further comparisons with MaxAggregation, MAMBA, MaxAggregation-SI.

Data collection. MAMBA uses roll-in, roll-out (RIRO) data collection. That is, they roll in with the student over a random horizon length, then roll out the remainder of the episode with an expert. SIO instead always acts in the environment with a policy that is itself training to maximize its returns (the selector). A downside of RIRO training is that, for infinite-horizon tasks, it requires estimating the empirical horizon length periodically throughout training. A benefit of RIRO training is that it might help with long-horizon exploration. In contrast, SIO may take a long time to get to parts of the state space where certain experts are best. In particular, this may happen if an expert happens to be bad near the initial state distribution and near other policies, but great in some far off states.

Value learning vs. policy learning. Unlike MaxAggregation, MaxAggregation-SI, and MAMBA, SIO learns about experts by policy gradient, rather than value learning. An algorithm similar to MAMBA can be implemented with one K-head or $K + 1$-head critic for $V_k(s)$, and one student actor. Our implementation of MaxAggregation-SI instead uses a separate critic for the student. We leave investigation of a combined critic for MaxAggregation to future work. MAMBA's critic plays a role similar to SIO's selector, though overall is somewhat incomparable to our implementation of SIO, where we use one single-head critic for $V(s)$ (shared between the selector and the student) and two actors. However, SIO's selector and student could also use Q-learning for the selector (with limited-size or no replay buffer to handle nonstationarity for the SI case), and/or Q-learning for the student (with BC handled by e.g. Hester et al. (2018)).

B.2 Additional Related Work

Robust Policy Improvement (RPI) (Liu et al., 2024) introduces the self-inclusive (SI) variants of MaxFollowing and MaxAggregation, which they call max$^+$-following and max$^+$-aggregation. Similar to Liu et al. (2023), they also introduce an uncertainty-based approach to choosing which policy, among the experts and student, to roll out during their RIRO training. A small downside is that the uncertainty component requires a value function ensemble for every expert and the student. In contrast, SIO learns an actor network to select among the experts and student during training. An important direction for future work is to compare RPI with SIO. Relatedly, Buckman et al. (2021) might suggest that, in a certain sense, an uncertainty-based approach in SIO might be better than our BC-based loss term.

AC-Teach (Kurenkov et al., 2020) uses a Bayesian critic with Thompson sampling and a commitment rule to choose among expert and student actions. No behavior cloning is used, though the DDPG critic target may use expert actions.

Kickstarting (Schmitt et al., 2018) chooses which experts to use per task, not per state. This might not make the best use of experts that are good at different states of one task. Further, Kickstarting has its student always act in the environment, and distills the expert policies at the states the student visits. This might be inefficient if exploration is difficult. We leave testing similar variants of SIO for future work.

Among many other contributions, Mignacco (2025) introduces and theoretically analyzes an algorithm similar to SIO-NoSI, with no exploration in primitive actions.

Q2RL (Dodeja et al., 2026) studies offline-to-online improvement from one pretrained BC policy. It estimates a static Q-function for the BC policy first, and then gates between BC and RL actions by comparing the static BC Q-value to an online RL Q-value. In contrast, SIO dynamically learns

where to use BC throughout training, learning to use BC even where an expert would only be useful if stitched with other policies.

Single-expert algorithms most similar to ours include LOKI (Cheng et al., 2019), which does a phase of IL then a phase of RL, and TGRL (Shenfeld et al., 2023), which uses a Lagrangian approach to adapt how much to learn from the expert at states the student encounters (somewhat similar to Kickstarting). In contrast, throughout training, SIO learns at every state which experts (or the student) should act in the environment, which also controls which experts (if any) the student should learn from. For additional related work, see Table 2 in Liu et al. (2024).

Hussing et al. (2024) introduces MaxIteration, competing with MaxFollowing, under weaker assumptions than MaxAggregation requires. Sengupta et al. (2025) introduces AdversarialMaxIteration (AMI), handling adversarial initial-state distributions.

C Further Experimental Results

Table 4: To alleviate concerns about averaging and outlier environments, here we compare *rankings* of mean returns over all {algorithm, environment, experts} combinations, ignoring CI overlaps.

Algorithm	Mean rank ↓	Beats MaxAgg-NoSI ↑	Beats MaxAgg-SI ↑
PPO	7.28 [7.03, 7.45]	0.15 [0.12, 0.23]	0.60 [0.50, 0.65]
MaxAgg-SI, $\lambda_{\mathrm{MA}} = 0.95$	7.50 [7.30, 7.70]	0.20 [0.10, 0.20]	—
MaxAgg-NoSI, $\lambda_{\mathrm{MA}} = 0.95$	5.25 [5.12, 5.45]	—	0.80 [0.80, 0.90]
SIO, $\xi = 0$	4.62 [4.38, 4.88]	0.65 [0.62, 0.70]	0.70 [0.70, 0.78]
SIO, $\xi = 0$, NoBC	8.32 [8.10, 8.47]	0.10 [0.05, 0.15]	0.28 [0.23, 0.35]
SIO, $\xi = 0$, $\beta = 0.5$	5.17 [4.92, 5.40]	0.62 [0.60, 0.68]	0.75 [0.70, 0.80]
SIO, $\xi = 0$, NoSI	5.05 [4.83, 5.20]	**0.68 [0.62, 0.70]**	0.72 [0.68, 0.75]
SIO, $\xi = 0.5$	4.70 [4.42, 4.92]	0.55 [0.53, 0.60]	0.85 [0.78, 0.85]
AMSIO, $\xi = 0.5$	4.15 [3.85, 4.35]	0.62 [0.62, 0.72]	**0.93 [0.85, 0.93]**
ISSIO, $\xi = 0.5$	**2.95 [2.90, 3.33]**	**0.68 [0.60, 0.70]**	0.88 [0.85, 0.90]

Table 5: Per-environment normalized returns, averaged over expert sets (PPO uses no experts).

Algorithm	`Acrobot-v1`	`CartPole-v1`	`MountainCar-v0`	`PendulumDiscrete-v1`
PPO	119.0 [112.1, 125.7]	121.9 [119.6, 124.2]	3.1 [0.6, 6.6]	-45.8 [-50.3, -41.5]
MaxAgg-SI, $\lambda_{\mathrm{MA}} = 0.95$	119.4 [117.2, 121.6]	121.0 [120.2, 121.9]	11.4 [10.0, 12.8]	-59.5 [-61.0, -57.9]
MaxAgg-NoSI, $\lambda_{\mathrm{MA}} = 0.95$	130.1 [128.5, 131.8]	123.8 [123.1, 124.6]	30.9 [29.2, 32.6]	-23.2 [-24.4, -22.0]
SIO, $\xi = 0$	143.1 [142.0, 144.2]	123.3 [122.2, 124.4]	**61.4 [58.1, 64.7]**	-6.4 [-7.7, -5.0]
SIO, $\xi = 0.5$	147.8 [146.6, 148.9]	133.1 [132.4, 133.7]	31.6 [28.7, 34.4]	-2.8 [-3.8, -1.7]
AMSIO, $\xi = 0.5$	**154.6 [153.5, 155.8]**	135.6 [135.1, 136.2]	21.2 [19.0, 23.5]	-0.5 [-1.6, 0.6]
ISSIO, $\xi = 0.5$	153.9 [152.8, 155.0]	**136.2 [135.7, 136.7]**	34.4 [31.3, 37.7]	**1.4 [0.3, 2.4]**

Table 6: Table 1 with all learning rates set to 10^{-3}.

Experts	MaxAgg-SI $\lambda_{\text{MA}} = 0.95$	MaxAgg-NoSI $\lambda_{\text{MA}} = 0.95$	SIO $\xi = 0.5$	AMSIO $\xi = 0.5$	ISSIO $\xi = 0.5$
R	67.1 [62.7, 71.0]	60.4 [58.1, 62.5]	72.7 [66.5, 79.1]	**81.3 [74.4, 88.5]**	76.8 [69.4, 84.2]
2R	66.6 [62.5, 70.7]	57.6 [52.1, 61.5]	72.5 [66.4, 78.6]	**76.4 [70.6, 82.7]**	75.3 [68.7, 82.3]
8R	65.1 [60.2, 69.2]	53.6 [48.1, 58.7]	72.1 [67.3, 77.6]	**74.8 [68.4, 81.8]**	65.0 [62.9, 67.2]
G	78.9 [68.3, 88.5]	86.2 [80.3, 92.1]	117.6 [114.2, 120.9]	108.2 [105.2, 111.0]	**118.0 [115.6, 120.5]**
G2R	87.5 [80.3, 94.5]	87.7 [79.8, 95.0]	**99.9 [91.8, 107.6]**	93.4 [85.5, 101.5]	98.8 [92.1, 105.1]
G7R	84.0 [78.1, 89.6]	78.8 [72.3, 84.7]	88.5 [81.9, 94.7]	87.4 [81.0, 93.6]	**98.6 [95.8, 101.6]**
Comp1	70.8 [63.2, 77.1]	89.7 [82.9, 96.4]	102.5 [95.1, 109.7]	97.5 [91.1, 103.0]	**105.8 [98.2, 112.4]**
Comp2	71.0 [66.2, 76.0]	71.1 [62.4, 79.4]	98.9 [93.0, 104.8]	100.4 [97.6, 102.7]	**107.4 [101.0, 112.4]**
Comp3	75.5 [70.0, 81.9]	94.2 [86.0, 101.5]	101.7 [95.4, 107.1]	93.3 [87.4, 98.6]	**106.0 [100.9, 110.2]**
Comp4	62.1 [56.6, 66.1]	62.8 [57.1, 68.1]	74.1 [67.0, 81.8]	**81.3 [75.6, 87.3]**	77.1 [70.4, 84.1]

Table 7: Table 4 with all learning rates set to 10^{-3}.

Algorithm	Mean rank ↓	Beats MaxAgg-NoSI ↑	Beats MaxAgg-SI ↑
PPO	5.90 [5.70, 6.08]	0.60 [0.55, 0.65]	0.78 [0.68, 0.85]
MaxAgg-SI, $\lambda_{\text{MA}} = 0.95$	7.15 [6.92, 7.40]	0.45 [0.38, 0.53]	—
MaxAgg-NoSI, $\lambda_{\text{MA}} = 0.95$	7.08 [6.78, 7.20]	—	0.50 [0.42, 0.55]
SIO, $\xi = 0$	4.97 [4.67, 5.25]	0.75 [0.68, 0.80]	0.65 [0.60, 0.75]
SIO, $\xi = 0.5$	4.04 [3.65, 4.30]	**0.88 [0.80, 0.90]**	0.82 [0.78, 0.90]
AMSIO, $\xi = 0.5$	4.12 [3.80, 4.38]	0.80 [0.72, 0.88]	**0.90 [0.85, 0.97]**
ISSIO, $\xi = 0.5$	**3.00 [2.80, 3.38]**	**0.88 [0.85, 0.90]**	0.85 [0.82, 0.93]

Table 8: Table 5 with all learning rates set to 10^{-3}.

Algorithm	Acrobot-v1	CartPole-v1	MountainCar-v0	PendulumDiscrete-v1
PPO	117.6 [113.5, 120.8]	123.2 [122.4, 123.9]	9.8 [5.0, 15.1]	59.1 [52.6, 65.1]
MaxAgg-SI, $\lambda_{\text{MA}} = 0.95$	110.1 [105.9, 113.6]	121.9 [121.5, 122.3]	21.2 [16.2, 26.1]	38.2 [33.6, 42.4]
MaxAgg-NoSI, $\lambda_{\text{MA}} = 0.95$	91.4 [89.8, 92.9]	114.1 [111.4, 116.7]	33.5 [28.0, 38.7]	57.9 [52.8, 62.9]
SIO, $\xi = 0$	98.1 [95.3, 100.8]	118.3 [114.9, 121.6]	45.5 [38.8, 52.5]	55.9 [51.5, 60.1]
SIO, $\xi = 0.5$	118.7 [117.3, 120.0]	124.8 [124.6, 125.0]	56.9 [49.7, 64.1]	**59.7 [56.6, 62.9]**
AMSIO, $\xi = 0.5$	**122.4 [121.9, 122.8]**	125.2 [125.0, 125.4]	58.9 [52.3, 65.3]	51.1 [47.3, 54.8]
ISSIO, $\xi = 0.5$	122.2 [121.1, 123.3]	**125.3 [125.1, 125.5]**	**66.0 [59.6, 72.3]**	58.0 [55.2, 60.9]

D Omitted Proofs

D.1 The Regret of the Behavior Policy

Proof of Theorem 1. By the performance difference lemma,

$$\begin{aligned}
J(\mu) - J(\mu_t) &= \mathbb{E}_{s\sim d^\mu}\left[\langle \mu(\cdot|s) - \mu_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle\right] \\
&= \mathbb{E}_{s\sim d^\mu}\Bigg[\sum_{k=1}^{K}\left(p_\mu(k|s) - p_t(k|s)\right)\langle \pi^{(k)}(\cdot|s), A^{\mu_t}(s,\cdot)\rangle \\
&\qquad + \left(p_\mu(K+1|s) - p_t(K+1|s)\right)\langle \pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle \\
&\qquad + p_\mu(K+1|s)\langle \mu_\perp(\cdot|s) - \pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle\Bigg] \\
&= \mathbb{E}_{s\sim d^\mu}\left[\sum_{k=1}^{K+1}\left(p_\mu(k|s) - p_t(k|s)\right)A_t^{\mu_t}(s,k)\right] \\
&\qquad + \mathbb{E}_{s\sim d^\mu}\left[p_\mu(K+1|s)\langle \mu_\perp(\cdot|s) - \pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle\right].
\end{aligned}$$

The statewise p-update is

$$p_{t+1}(\cdot|s) = \operatorname*{argmax}_{p(\cdot|s)\in\Delta_{K+1}}\left\{\sum_{k=1}^{K+1} p(k|s)A_t^{\mu_t}(s,k) - \eta^{-1}D_{\mathrm{KL}}(p(\cdot|s)\|p_t(\cdot|s))\right\}.$$

Applying Lemma 7 with $\tau = 0$, $\eta = \eta$, and $r_t(k) = A_t^{\mu_t}(s,k)$ gives

$$\sum_{k=1}^{K+1}\left(p_\mu(k|s) - p_t(k|s)\right)A_t^{\mu_t}(s,k) \le 2L^2\eta + \eta^{-1}\left(D_{\mathrm{KL}}(p_\mu(\cdot|s)\|p_t(\cdot|s)) - D_{\mathrm{KL}}(p_\mu(\cdot|s)\|p_{t+1}(\cdot|s))\right).$$

Summing over t and using $p_1(\cdot|s) = \mathbf{1}/(K+1)$ gives

$$\sum_{t=1}^{T}\sum_{k=1}^{K+1}\left(p_\mu(k|s) - p_t(k|s)\right)A_t^{\mu_t}(s,k) \le 2L^2T\eta + \eta^{-1}\log(K+1).$$

Combining this bound with the residual primitive term proves the theorem. □

D.2 The Regret of the Student Policy

Throughout this subsection, we write

$$q_t(s) := p_t(K+1|s). \tag{3}$$

When $q_t(s) < 1$, define

$$\overline{\mu}_t(\cdot|s) := \frac{\sum_{k=1}^{K} p_t(k|s)\pi^{(k)}(\cdot|s)}{1 - q_t(s)}.$$

When $q_t(s) = 1$, define $\overline{\mu}_t(\cdot|s)$ arbitrarily, since every occurrence of $\overline{\mu}_t$ is multiplied by $1 - q_t(s)$.

Assumption 1 (Optimal-action gap)**.** *There exists $\Delta_{\min} \in (0,1]$ such that, for every s and every $a \neq \mu^\star(s)$,*

$$V^{\mu^\star}(s) - Q^{\mu^\star}(s,a) \ge \Delta_{\min}.$$

Assumption 2 (Concentrability)**.** *There exists $C \ge 1$ such that, for any initial distribution $\varrho \in \Delta_{\mathcal{S}}$ and any non-stationary policies ν_1, ν_2,*

$$\max_{s\in\mathcal{S}} \frac{d_\varrho^{\nu_1}(s)}{d_\varrho^{\nu_2}(s)} \le C.$$

Assumption 3 (Behavior-cloning weight). *For some $\zeta \in (0, 1/2)$, the behavior-cloning weight is*

$$\beta_t(s) = q_t(s)t^{-\zeta}.$$

We give a high-level proof sketch first.

Proof sketch of Theorem 2. Our proof goes through three main steps outlined below. For simplicity, the explanation below omits the state dependence.

1. (Lemma 4) Based on the primitive update rule and a proper choice of $\beta_t(s)$, we can bound $\mathrm{Reg}_{\mathrm{stu}}(\mu^\star; s) \le o(T)$. Combining this with Theorem 1 allows us to obtain $\sum_t (J(\mu^\star) - J(\mu_t)) \le o(T)$. With the gap assumption, this implies $\sum_t \|\mu^\star - \mu_t\| = o(T)$.
2. The primitive objective roughly maximizes $\langle \pi, A^{\mu_t} \rangle - \lambda_t D_{\mathrm{KL}}(\mu_t \| \pi)$ for some $\lambda_t > 0$, where the first term corresponds to RL and the second term corresponds to BC. The regret guarantee implies $\sum_t (\langle \pi_t, A^{\mu_t} \rangle - \lambda_t D_{\mathrm{KL}}(\mu_t \| \pi_t)) \gtrsim \sum_t (\langle \mu^\star, A^{\mu_t} \rangle - \lambda_t D_{\mathrm{KL}}(\mu_t \| \mu^\star))$. On the other hand, $\sum_t \langle \mu^\star - \pi_t, A^{\mu_t} \rangle \gtrsim \sum_t \langle \mu^\star - \pi_t, A^{\mu^\star} \rangle - O\left(\sum_t \|\mu^\star - \mu_t\|\right) \ge 0 - o(T)$ by Step 1 and the property of $A^{\mu^\star}$. The two inequalities combined yields $\sum_t \lambda_t D_{\mathrm{KL}}(\mu_t \| \pi_t) \lesssim \sum_t \lambda_t D_{\mathrm{KL}}(\mu_t \| \mu^\star) + o(T)$.
3. With suitable λ_t (which is also achieved by properly choosing $\beta_t(s)$), $\sum_t \lambda_t D_{\mathrm{KL}}(\mu_t \| \pi_t) = o(T)$ further implies $\sum_t \|\mu_t - \pi_t\| = o(T)$ and thus $\sum_t (J(\mu_t) - J(\pi_t)) = o(T)$. Combining Step 1 and Step 3, we get $\sum_t (J(\mu^\star) - J(\pi_t)) = o(T)$.

□

Define

$$M_\zeta := \left\lceil \left(\frac{12C}{\Delta_{\min}} \right)^{1/\zeta} \right\rceil.$$

Lemma 1 (Uniform non-stationary concentrability implies advantage stability). *Assume rewards are bounded in $[0, 1]$. Under Assumption 2, for every policy μ,*

$$\mathbb{E}_{s \sim d^{\mu^\star}} \left[\|A^\mu(s, \cdot) - A^{\mu^\star}(s, \cdot)\|_\infty \right] \le 2CL \mathbb{E}_{s \sim d^{\mu^\star}} \left[\|\mu(\cdot|s) - \mu^\star(\cdot|s)\|_1 \right].$$

Proof.

$$\begin{aligned}
&\mathbb{E}_{s \sim d^{\mu^\star}} \left[\left| A^\mu(s, a) - A^{\mu^\star}(s, a) \right| \right] \\
&\le \mathbb{E}_{s \sim d^{\mu^\star}} \left[\left| Q^\mu(s, a) - Q^{\mu^\star}(s, a) \right| + \left| V^\mu(s) - V^{\mu^\star}(s) \right| \right] \\
&\le L \mathbb{E}_{s \sim d^{\mu^\star}} \left[\mathbb{E}_{s' \sim d^{\mu^\star}_{s,a}} \left[\|\mu(\cdot|s') - \mu^\star(\cdot|s')\|_1 \right] + \mathbb{E}_{s' \sim d^{\mu^\star}_s} \left[\|\mu(\cdot|s') - \mu^\star(\cdot|s')\|_1 \right] \right] \\
&\hspace{20em} \text{(by performance difference lemma)} \\
&\le 2CL \mathbb{E}_{s \sim d^{\mu^\star}} \left[\|\mu(\cdot|s) - \mu^\star(\cdot|s)\|_1 \right]. \qquad \text{(by the definition of } C\text{)}
\end{aligned}$$

□

Lemma 2 (Primitive regret of the learner update). *For any policy $\pi : \mathcal{S} \to \Delta_{\mathcal{A}}$ and any state s,*

$$\begin{aligned}
\mathrm{Reg}_{\mathrm{stu}}(\pi; s) &\le 2L^2 T \eta + \eta^{-1} D_{\mathrm{KL}}(\pi(\cdot|s) \| \pi_1(\cdot|s)) \\
&\quad + \sum_{t=1}^{T} \tau_t(s) \left(6 D_{\mathrm{H}}^2(\pi(\cdot|s) \| \overline{\mu}_t(\cdot|s)) - \frac{1}{1 + 2\eta \tau_t(s)} D_{\mathrm{H}}^2(\pi_t(\cdot|s) \| \overline{\mu}_t(\cdot|s)) \right),
\end{aligned}$$

where $\tau_t(s) = t^{-\zeta}(1 - q_t(s))$.

Proof. Fix s and define

$$r_t(s,\cdot) := A^{\mu_t}(s,\cdot).$$

Since the KL update keeps full support when initialized with full support, $q_t(s) > 0$. Dividing the statewise learner update by $q_t(s)$ gives

$$\pi_{t+1}(\cdot|s) = \operatorname*{argmax}_{\pi'(\cdot|s)} \left\{ \langle \pi'(\cdot|s), r_t(s,\cdot)\rangle - \eta^{-1} D_{\mathrm{KL}}(\pi'(\cdot|s)\|\pi_t(\cdot|s)) - \tau_t(s) D_{\mathrm{KL}}(\overline{\mu}_t(\cdot|s)\|\pi'(\cdot|s)) \right\}$$

where we use the fact that

$$\begin{aligned}
&\frac{\beta_t(s)}{q_t(s)} \sum_{k=1}^{K} p_t(k|s) \mathbb{E}_{a\sim\pi^{(k)}(\cdot|s)}[\log \pi'(a|s)] \\
&= t^{-\zeta}(1-q_t(s)) \mathbb{E}_{a\sim\overline{\mu}_t(\cdot|s)}[\log \pi'(a|s)] \\
&\qquad\qquad \text{(by Assumption 3 and the definition of } q_t(s), \overline{\mu}_t(\cdot|s) \text{ in (3))} \\
&= -t^{-\zeta}(1-q_t(s)) D_{\mathrm{KL}}(\overline{\mu}_t(\cdot|s)\|\pi'(a|s)) + t^{-\zeta}(1-q_t(s)) \mathbb{E}_{a\sim\overline{\mu}_t(\cdot|s)}[\log \overline{\mu}_t(a|s)]
\end{aligned}$$

and the last term above is a constant w.r.t. π'.

Apply Lemma 7 with

$$x_t = \pi_t(\cdot|s), \qquad y_t = \overline{\mu}_t(\cdot|s), \qquad u = \pi(\cdot|s), \qquad r_t = r_t(s,\cdot), \qquad \eta = \eta, \qquad \tau = \tau_t(s).$$

Summing the resulting inequality over t and dropping the final nonnegative KL term proves the claim. □

Lemma 3 (The learner branch is selected on states where experts miss the optimal action)**.** *Under Assumption 1, 2, 3,*

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}}\left[y(s)(1-q_t(s))\right] \le O\left(\frac{C}{\Delta_{\min}}(B_{\exp}(T) + M_\zeta)\right) = O(B_{\mathrm{opt}}(T)).$$

Proof. Apply Theorem 1 to the comparator that chooses $k^\star(s)$ when $y(s) = 0$ and chooses the learner branch with residual policy $\mu^\star$ when $y(s) = 1$. This gives

$$\sum_{t=1}^{T} (J(\mu^\star) - J(\mu_t)) \le B_{\exp}(T) + \mathbb{E}_{d^{\mu^\star}}\left[y(s)\mathrm{Reg}_{\mathrm{stu}}(\mu^\star; s)\right].$$

By Lemma 2 with $\pi = \mu^\star$,

$$\mathrm{Reg}_{\mathrm{stu}}(\mu^\star; s) \le B_{\exp}(T) + 6\sum_{t=1}^{T} t^{-\zeta}(1-q_t(s)) D^2_{\mathrm{H}}(\mu^\star(\cdot|s)\|\overline{\mu}_t(\cdot|s)),$$

where we dropped the negative Hellinger term. Since $D^2_{\mathrm{H}}(\cdot\|\cdot) \le 1$,

$$\mathbb{E}_{d^{\mu^\star}}\left[y(s)\mathrm{Reg}_{\mathrm{stu}}(\mu^\star; s)\right] \le B_{\exp}(T) + 6\sum_{t=1}^{T} t^{-\zeta} \mathbb{E}_{d^{\mu^\star}}\left[y(s)(1-q_t(s))\right].$$

Therefore,

$$\sum_{t=1}^{T} (J(\mu^\star) - J(\mu_t)) \le O(B_{\exp}(T)) + 6\sum_{t=1}^{T} t^{-\zeta} \mathbb{E}_{d^{\mu^\star}}\left[y(s)(1-q_t(s))\right].$$

On the other hand, by the performance difference lemma, the optimal-action gap, and Assumption 2,

$$J(\mu^\star) - J(\mu_t) \ge \frac{\Delta_{\min}}{C} \mathbb{E}_{d^{\mu^\star}}\left[1 - \mu_t(a^\star(s)|s)\right].$$

On states with $y(s) = 1$, no expert chooses $a^\star(s)$, so $1 - \mu_t(a^\star(s)|s) \geq 1 - q_t(s)$. Thus

$$\frac{\Delta_{\min}}{C} \sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[y(s)(1 - q_t(s)) \right] \leq O(B_{\exp}(T)) + 6 \sum_{t=1}^{T} t^{-\zeta} \mathbb{E}_{d^{\mu^\star}} \left[y(s)(1 - q_t(s)) \right].$$

The claim follows from Lemma 9 with

$$a_t = \mathbb{E}_{d^{\mu^\star}} \left[y(s)(1 - q_t(s)) \right], \qquad c = \frac{\Delta_{\min}}{C}, \qquad b = 6, \qquad B = O(B_{\exp}(T)).$$

□

Lemma 4 (Behavior policy is close to $\mu^\star$)**.** *Under the assumptions of Lemma 3, the following hold:*

$$\sum_{t=1}^{T} \left(J(\mu^\star) - J(\mu_t) \right) \leq O\left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T) \right),$$

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[\|\mu_t(\cdot|s) - \mu^\star(\cdot|s)\|_1 \right] \leq O\left(\frac{C}{\Delta_{\min}} \left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T) \right) \right)$$

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[\|A^{\mu_t}(s,\cdot) - A^{\mu^\star}(s,\cdot)\|_\infty \right] \leq O\left(\frac{C^2 L}{\Delta_{\min}} \left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T) \right) \right).$$

Proof. Define

$$y(s) = \mathbf{1}\left[\mu^\star(s) \notin \{\pi^{(k)}(s)\}_{k=1}^{K} \right].$$

The proof of Lemma 3 showed that

$$\sum_{t=1}^{T} \left(J(\mu^\star) - J(\mu_t) \right) \leq O(B_{\exp}(T)) + 6 \sum_{t=1}^{T} t^{-\zeta} \mathbb{E}_{d^{\mu^\star}} \left[y(s)(1 - q_t(s)) \right],$$

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[y(s)(1 - q_t(s)) \right] \leq O\left(B_{\mathrm{opt}}(T) \right).$$

Combining them and using the facts that $y(s) \leq \sigma$ and $t^{-\zeta} \leq 1$, we get

$$\sum_{t=1}^{T} \left(J(\mu^\star) - J(\mu_t) \right) \leq O\left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T) \right).$$

For the second claim, the optimal-action gap and Assumption 2 give

$$J(\mu^\star) - J(\mu_t) \geq \frac{\Delta_{\min}}{2C} \mathbb{E}_{d^{\mu^\star}} \left[\|\mu_t(\cdot|s) - \mu^\star(\cdot|s)\|_1 \right],$$

because $\mu^\star$ is deterministic and $\|\mu_t(\cdot|s) - \mu^\star(\cdot|s)\|_1 = 2(1 - \mu_t(a^\star(s)|s))$. Summing over t and using the first claim gives

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[\|\mu_t(\cdot|s) - \mu^\star(\cdot|s)\|_1 \right] \leq O\left(\frac{C}{\Delta_{\min}} \left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T) \right) \right).$$

For the last claim, by Lemma 1,

$$\mathbb{E}_{d^{\mu^\star}} \left[\|A^{\mu_t}(s,\cdot) - A^{\mu^\star}(s,\cdot)\|_\infty \right] \leq 2CL \mathbb{E}_{d^{\mu^\star}} \left[\|\mu_t(\cdot|s) - \mu^\star(\cdot|s)\|_1 \right].$$

Summing over t and applying Lemma 4 proves the claim. □

Lemma 5 ($\bar{\mu}_t$ is close to $\mu^\star$)**.** *Under the previous assumptions,*

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[(1-q_t(s)) D_H^2(\mu^\star(\cdot|s)\|\bar{\mu}_t(\cdot|s))\right] \le O\left(\frac{C}{\Delta_{\min}}\left(B_{\exp}(T)+\sigma B_{\mathrm{opt}}(T)\right)\right).$$

Proof. For every state s,

$$(1-q_t(s)) D_H^2(\mu^\star(\cdot|s)\|\bar{\mu}_t(\cdot|s)) \le 1-\mu_t(\mu^\star(s)|s).$$

Indeed, if $y(s)=0$, then this follows because the expert mixture disagrees with $\mu^\star$ only through the expert mass assigned to suboptimal experts. If $y(s)=1$, then no expert chooses $\mu^\star(s)$, so

$$1-\mu_t(\mu^\star(s)|s) \ge 1-q_t(s),$$

while $D_H^2(\mu^\star(\cdot|s)\|\bar{\mu}_t(\cdot|s)) \le 1$.

By the optimal-action gap and concentrability,

$$J(\mu^\star)-J(\mu_t) \ge \frac{\Delta_{\min}}{C}\mathbb{E}_{d^{\mu^\star}}\left[1-\mu_t(\mu^\star(s)|s)\right].$$

Therefore,

$$\mathbb{E}_{d^{\mu^\star}}\left[(1-q_t(s)) D_H^2(\mu^\star(\cdot|s)\|\bar{\mu}_t(\cdot|s))\right] \le \frac{C}{\Delta_{\min}}\left(J(\mu^\star)-J(\mu_t)\right).$$

Summing over t and applying Lemma 4 proves the claim. □

Lemma 6 (Decayed BC control of the learner)**.** *Under the previous assumptions,*

$$\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}} \left[t^{-\zeta}(1-q_t(s)) D_{\mathrm{H}}^2(\pi_t(\cdot|s)\|\overline{\mu}_t(\cdot|s))\right] \le O\left(\frac{C^2 L}{\Delta_{\min}}\left(B_{\exp}(T)+\sigma B_{\mathrm{opt}}(T)\right)\right).$$

Proof. Apply Lemma 7 to the learner update with

$$u=\mu^\star(\cdot|s), \quad x_t=\pi_t(\cdot|s), \quad y_t=\overline{\mu}_t(\cdot|s), \quad r_t=A^{\mu_t}(s,\cdot), \quad \eta=\eta, \quad \tau=\tau_t(s).$$

This gives an upper bound:

$$\begin{aligned}
&\langle \mu^\star(\cdot|s)-\pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle \\
&\le 2L^2\eta+\eta^{-1}\left[D_{\mathrm{KL}}(\mu^\star(\cdot|s)\|\pi_t(\cdot|s))-D_{\mathrm{KL}}(\mu^\star(\cdot|s)\|\pi_{t+1}(\cdot|s))\right] \\
&\qquad + 6\tau_t(s) D_{\mathrm{H}}^2(\mu^\star(\cdot|s)\|\overline{\mu}_t(\cdot|s)) - \tau_t(s)\frac{1}{1+2\eta\tau_t(s)} D_{\mathrm{H}}^2(\pi_t(\cdot|s)\|\overline{\mu}_t(\cdot|s))
\end{aligned}$$

On the other hand, the optimal-action gap and advantage perturbation imply

$$\begin{aligned}
\langle \mu^\star(\cdot|s)-\pi_t(\cdot|s), A^{\mu_t}(s,\cdot)\rangle &\ge \langle \mu^\star(\cdot|s)-\pi_t(\cdot|s), A^{\mu^\star}(s,\cdot)\rangle - 2\|A^{\mu_t}(s,\cdot)-A^{\mu^\star}(s,\cdot)\|_\infty \\
&\ge -2\|A^{\mu_t}(s,\cdot)-A^{\mu^\star}(s,\cdot)\|_\infty.
\end{aligned}$$

Combining the two inequalities and using $\frac{1}{1+2\eta\tau_t(s)} \ge \frac{1}{3}$ due to $0 \le \tau_t(s) \le 1$ and $\eta \le 1$ yields

$$\begin{aligned}
\frac{1}{3}\tau_t(s) D_{\mathrm{H}}^2(\pi_t(\cdot|s)\|\overline{\mu}_t(\cdot|s)) &\le 2L^2\eta+\eta^{-1}\left[D_{\mathrm{KL}}(\mu^\star(\cdot|s)\|\pi_t(\cdot|s))-D_{\mathrm{KL}}(\mu^\star(\cdot|s)\|\pi_{t+1}(\cdot|s))\right] \\
&\qquad + 6\tau_t(s) D_{\mathrm{H}}^2(\mu^\star(\cdot|s)\|\overline{\mu}_t(\cdot|s)) + 2\|A^{\mu_t}(s,\cdot)-A^{\mu^\star}(s,\cdot)\|_\infty.
\end{aligned}$$

Using $\tau_t(s) = t^{-\zeta}(1 - q_t(s))$, taking expectation under $d^{\mu^\star}$, and summing over t gives

$$\begin{aligned}
&\frac{1}{3}\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}}\left[t^{-\zeta}(1-q_t(s))D_{\mathrm{H}}^2(\pi_t(\cdot|s)\|\overline{\mu}_t(\cdot|s))\right] \\
&\le B_{\exp}(T) + 6\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}}\left[t^{-\zeta}(1-q_t(s))D_{\mathrm{H}}^2(\mu^\star(\cdot|s)\|\overline{\mu}_t(\cdot|s))\right] \\
&\quad + 2\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}}\left[\|A^{\mu_t}(s,\cdot) - A^{\mu^\star}(s,\cdot)\|_\infty\right].
\end{aligned}$$

Since $t^{-\zeta} \le 1$, Lemma 5 bounds the middle term by

$$O\left(\frac{C}{\Delta_{\min}}\left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T)\right)\right).$$

Lemma 4 bounds the last term by

$$O\left(\frac{C^2 L}{\Delta_{\min}}\left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T)\right)\right).$$

Combining terms proves the claim. □

Theorem 3 (Regret of the deployed learner)**.** *Under Assumption 1, 2, 3,*

$$\sum_{t=1}^{T}\left(J(\mu^\star) - J(\pi_t)\right) \le O\left(CLT^{(1+\zeta)/2}\sqrt{\frac{C^2 L}{\Delta_{\min}}\left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T)\right)}\right).$$

Proof. For every t,

$$J(\mu^\star) - J(\pi_t) = J(\mu^\star) - J(\mu_t) + J(\mu_t) - J(\pi_t).$$

By Lemma 4,

$$\sum_{t=1}^{T}\left(J(\mu^\star) - J(\mu_t)\right) \le O\left(B_{\exp}(T) + \sigma B_{\mathrm{opt}}(T)\right).$$

It remains to control

$$\sum_{t=1}^{T}\left(J(\mu_t) - J(\pi_t)\right).$$

By the performance difference lemma, boundedness of the advantage, and Assumption 2,

$$J(\mu_t) - J(\pi_t) \le CL\mathbb{E}_{d^{\mu^\star}}\left[\|\mu_t(\cdot|s) - \pi_t(\cdot|s)\|_1\right].$$

Since

$$\mu_t(\cdot|s) = (1 - q_t(s))\overline{\mu}_t(\cdot|s) + q_t(s)\pi_t(\cdot|s),$$

we have

$$\|\mu_t(\cdot|s) - \pi_t(\cdot|s)\|_1 = (1 - q_t(s))\|\overline{\mu}_t(\cdot|s) - \pi_t(\cdot|s)\|_1.$$

Split this term according to $y(s)$.

On the bad-expert states, $\|\overline{\mu}_t(\cdot|s) - \pi_t(\cdot|s)\|_1 \le 2$, so Lemma 3 gives

$$\sum_{t=1}^{T}\mathbb{E}_{d^{\mu^\star}}\left[y(s)(1-q_t(s))\|\overline{\mu}_t(\cdot|s) - \pi_t(\cdot|s)\|_1\right] \le O\left(B_{\mathrm{opt}}(T)\right).$$

After multiplying by CL, this contributes $O\left(CLB_{\mathrm{opt}}(T)\right)$.

On the good-expert states, use $\|p-q\|_1 \le 2\sqrt{2}D_{\mathrm{H}}(p\|q)$ and Cauchy–Schwarz:

$$\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}}\left[(1-y(s))(1-q_t(s))\|\overline{\mu}_t(\cdot|s)-\pi_t(\cdot|s)\|_1\right] \\
&\le 2\sqrt{2}\sqrt{\sum_{t=1}^{T} t^{\zeta}\mathbb{E}_{d^{\mu^\star}}\left[(1-y(s))(1-q_t(s))\right]} \\
&\quad \cdot \sqrt{\sum_{t=1}^{T} \mathbb{E}_{d^{\mu^\star}}\left[(1-y(s))t^{-\zeta}(1-q_t(s))D_{\mathrm{H}}^2(\pi_t(\cdot|s)\|\overline{\mu}_t(\cdot|s))\right]}.
\end{aligned}$$

The first factor is at most $T^{(1+\zeta)/2}$, and the second factor is bounded by Lemma 6. Therefore, after multiplying by the leading factor CL, the good-expert contribution is at most

$$O\left(CLT^{(1+\zeta)/2}\sqrt{\frac{C^2L}{\Delta_{\min}}\left(B_{\exp}(T)+\sigma B_{\mathrm{opt}}(T)\right)}\right).$$

Combining the behavior-policy regret, the bad-expert deployment term, and the good-expert deployment term proves the theorem. □

D.3 Auxiliary Lemmas

Lemma 7 (General ascent lemma)**.** *Let $m \in \mathbb{N}$, $x_t, y_t \in \Delta_m$, $r_t \in [-L,L]^m$, $\eta > 0$, and $\tau \ge 0$. Suppose*

$$x_{t+1} = \operatorname*{argmax}_{x\in\Delta_m}\left\{\langle x, r_t\rangle - \eta^{-1}D_{\mathrm{KL}}(x\|x_t) - \tau D_{\mathrm{KL}}(y_t\|x)\right\}.$$

Then for every $u \in \Delta_m$,

$$\langle u-x_t, r_t\rangle \le 2L^2\eta + \eta^{-1}\left(D_{\mathrm{KL}}(u\|x_t) - D_{\mathrm{KL}}(u\|x_{t+1})\right) + \tau\left(6D_{\mathrm{H}}^2(u\|y_t) - \frac{1}{1+2\eta\tau}D_{\mathrm{H}}^2(x_t\|y_t)\right).$$

Proof. By the first-order optimality condition for x_{t+1}, for every $u \in \Delta_m$,

$$\begin{aligned}
0 &\ge (u-x_{t+1})^\top \nabla_x\left(\langle x, r_t\rangle - \eta^{-1}D_{\mathrm{KL}}(x\|x_t) - \tau D_{\mathrm{KL}}(y_t\|x)\right)\Big|_{x=x_{t+1}} \\
&= \langle u-x_{t+1}, r_t\rangle + \eta^{-1}\left(D_{\mathrm{KL}}(u\|x_{t+1}) - D_{\mathrm{KL}}(u\|x_t) + D_{\mathrm{KL}}(x_{t+1}\|x_t)\right) + \tau\left(\sum_{i=1}^{m}\frac{u(i)y_t(i)}{x_{t+1}(i)} - 1\right).
\end{aligned}$$

By Lemma 8,

$$\sum_{i=1}^{m}\frac{u(i)y_t(i)}{x_{t+1}(i)} - 1 \ge D_{\mathrm{H}}^2(x_{t+1}\|y_t) - 6D_{\mathrm{H}}^2(u\|y_t).$$

Therefore,

$$\begin{aligned}
\langle u-x_t, r_t\rangle \le{}& \eta^{-1}\left(D_{\mathrm{KL}}(u\|x_t) - D_{\mathrm{KL}}(u\|x_{t+1}) - D_{\mathrm{KL}}(x_{t+1}\|x_t)\right) \\
&+ \tau\left(6D_{\mathrm{H}}^2(u\|y_t) - D_{\mathrm{H}}^2(x_{t+1}\|y_t)\right) + \langle x_{t+1}-x_t, r_t\rangle.
\end{aligned}$$

Using

$$\langle x'-x, r\rangle - \frac{1}{2\eta}D_{\mathrm{KL}}(x'\|x) \le 2L^2\eta,$$

we get

$$\langle u - x_t, r_t \rangle \le 2L^2\eta + \eta^{-1}\left(D_{\mathrm{KL}}(u\|x_t) - D_{\mathrm{KL}}(u\|x_{t+1})\right) - \frac{1}{2\eta} D_{\mathrm{KL}}(x_{t+1}\|x_t) + \tau\left(6D_{\mathrm{H}}^2(u\|y_t) - D_{\mathrm{H}}^2(x_{t+1}\|y_t)\right).$$

By the triangle inequality for Hellinger distance, for any $\nu > 0$,

$$D_{\mathrm{H}}(x_t\|y_t) \le D_{\mathrm{H}}(x_t\|x_{t+1}) + D_{\mathrm{H}}(x_{t+1}\|y_t).$$

Hence

$$D_{\mathrm{H}}^2(x_t\|y_t) \le (1+\nu) D_{\mathrm{H}}^2(x_{t+1}\|y_t) + \left(1 + \frac{1}{\nu}\right) D_{\mathrm{H}}^2(x_t\|x_{t+1}).$$

Rearranging and using $D_{\mathrm{H}}^2(x_t\|x_{t+1}) \le D_{\mathrm{KL}}(x_{t+1}\|x_t)$ gives

$$-D_{\mathrm{H}}^2(x_{t+1}\|y_t) \le -\frac{1}{1+\nu} D_{\mathrm{H}}^2(x_t\|y_t) + \frac{1}{\nu} D_{\mathrm{KL}}(x_{t+1}\|x_t).$$

Therefore,

$$\langle u - x_t, r_t \rangle \le 2L^2\eta + \eta^{-1}\left(D_{\mathrm{KL}}(u\|x_t) - D_{\mathrm{KL}}(u\|x_{t+1})\right) - \left(\frac{1}{2\eta} - \frac{\tau}{\nu}\right) D_{\mathrm{KL}}(x_{t+1}\|x_t) + \tau\left(6D_{\mathrm{H}}^2(u\|y_t) - \frac{1}{1+\nu} D_{\mathrm{H}}^2(x_t\|y_t)\right).$$

Choosing $\nu = 2\eta\tau$ gives

$$\frac{1}{1+\nu} = \frac{1}{1+2\eta\tau}.$$

Dropping the nonpositive KL term proves the lemma. □

Lemma 8. *Let $u, x, y \in \Delta_m$. It holds that*

$$\sum_{i=1}^{m} \frac{u(i)y(i)}{x(i)} - 1 \ge D_{\mathrm{H}}^2(x\|y) - 6D_{\mathrm{H}}^2(u\|y).$$

Proof. Let

$$r_i = \sqrt{\frac{y(i)}{x(i)}}, \qquad s_i = \sqrt{\frac{u(i)}{x(i)}}, \qquad d_i = s_i - r_i.$$

Then

$$\sum_i x(i) r_i^2 = \sum_i y(i) = 1, \qquad \sum_i x(i) s_i^2 = \sum_i u(i) = 1.$$

In particular,

$$\sum_i x(i)(r_i^2 - 1) = 0, \qquad \sum_i x(i)(2r_i d_i + d_i^2) = 0.$$

Using

$$D_{\mathrm{H}}^2(x\|y) = \sum_i x(i)(r_i - 1)^2, \qquad D_{\mathrm{H}}^2(u\|y) = \sum_i x(i) d_i^2,$$

we have

$$\sum_i \frac{u(i)y(i)}{x(i)} - 1 - D_{\mathrm{H}}^2(x\|y) + 6D_{\mathrm{H}}^2(u\|y) = \sum_i x(i)\Big[r_i^4 + 2r_i^3 d_i + r_i^2 d_i^2 - 1 - (r_i - 1)^2 + 6d_i^2\Big].$$

Now use the two zero identities above to rewrite the right-hand side as

$$\sum_i x(i)\Big[(r_i-1)^2 r_i(r_i+2) + 2r_i(r_i^2-1)d_i + (r_i^2+5)d_i^2\Big].$$

For each fixed $r_i \geq 0$, the expression inside the bracket is a quadratic in d_i with leading coefficient $r_i^2 + 5 > 0$. Its discriminant divided by 4 is

$$r_i^2(r_i^2-1)^2 - (r_i^2+5)(r_i-1)^2 r_i(r_i+2) = -(r_i-1)^2 r_i(4r_i+10) \leq 0.$$

Hence every summand is nonnegative. Therefore

$$\sum_i \frac{u(i)y(i)}{x(i)} - 1 \geq D_{\mathrm{H}}^2(x\|y) - 6D_{\mathrm{H}}^2(u\|y),$$

as desired. □

Lemma 9 (Absorption lemma)**.** *Let $a_1, \ldots, a_T \in [0,1]$. Suppose that, for some $c > 0$, $b > 0$, and $B \geq 0$,*

$$c\sum_{t=1}^{T} a_t \leq B + b\sum_{t=1}^{T} t^{-\zeta} a_t.$$

Let

$$M := \left\lceil \left(\frac{2b}{c}\right)^{1/\zeta} \right\rceil.$$

Then

$$\sum_{t=1}^{T} a_t \leq \frac{2B}{c} + \left(1 + \frac{2b}{c}\right) M.$$

Proof. Let $S_1 = \sum_{t=1}^{M} a_t$ and $S_2 = \sum_{t=M+1}^{T} a_t$. Since $a_t \leq 1$,

$$S_1 \leq M.$$

By the definition of M, for every $t > M$,

$$bt^{-\zeta} \leq \frac{c}{2}.$$

Therefore,

$$c(S_1 + S_2) \leq B + bS_1 + \frac{c}{2}S_2.$$

Rearranging gives

$$\frac{c}{2}S_2 \leq B + bS_1 \leq B + bM.$$

Thus

$$S_2 \leq \frac{2B}{c} + \frac{2bM}{c}.$$

Hence

$$\sum_{t=1}^{T} a_t = S_1 + S_2 \leq M + \frac{2B}{c} + \frac{2bM}{c} = \frac{2B}{c} + \left(1 + \frac{2b}{c}\right) M.$$

□

E Further Experiment Details

`PendulumDiscrete-v1` is the classic `Pendulum-v1` environment, but with the continuous torque action space replaced by three discrete actions applying torques -2, 0, and 2.

E.1 Hyperparameters

For all algorithms, we follow the PPO conventions and default hyperparameters of cleanRL (Huang et al., 2022; CleanRL contributors, 2026b), except that: for simplicity we do not include entropy or learning rate annealing; we sweep the actor and critic learning rates over a fixed grid; and for computational throughput, instead of CleanRL's default collection of 128 steps in each of four parallel environments, we use one environment with 512 steps per PPO rollout, preserving the default effective PPO batch size of 512 transitions per update. Using default hyperparameters (four parallel environments, 128 steps per PPO rollout, 10^{-3} learning rates), Table 9, Table 10, and Table 11 show our three best algorithms ($\xi = 0.5$) match or outperform both MaxAggregation and MaxAggregation-SI over almost all environment-averaged or expert-set-averaged results — and with even higher win-rates over all individual environment–experts combinations compared to the single-environment Table 7 setting.

Table 9: Table 6 with default hyperparameters (four parallel environments, 128 steps per PPO rollout, and all learning rates set to 10^{-3}).

Experts	MaxAgg-SI $\lambda_{\mathrm{MA}} = 0.95$	MaxAgg-NoSI $\lambda_{\mathrm{MA}} = 0.95$	SIO $\xi = 0.5$	AMSIO $\xi = 0.5$	ISSIO $\xi = 0.5$
R	47.2 [40.6, 53.9]	58.5 [55.4, 61.2]	62.0 [56.7, 67.3]	**74.2 [68.4, 79.4]**	66.0 [60.8, 71.4]
2R	61.8 [58.4, 65.5]	59.0 [55.5, 62.2]	60.5 [55.7, 65.5]	**69.5 [64.7, 74.4]**	58.3 [53.0, 64.0]
8R	55.8 [50.7, 60.0]	54.7 [50.7, 58.3]	52.0 [47.5, 56.6]	**64.9 [59.9, 70.3]**	49.2 [44.7, 53.7]
G	71.7 [65.2, 77.9]	71.4 [66.0, 77.6]	**139.2 [128.8, 148.1]**	112.9 [102.5, 124.0]	137.2 [127.2, 146.4]
G2R	68.5 [61.5, 74.7]	79.2 [73.2, 86.1]	**110.2 [95.3, 125.9]**	88.6 [78.8, 99.9]	107.9 [94.9, 122.6]
G7R	71.1 [65.7, 76.7]	79.2 [70.6, 86.7]	**96.8 [81.0, 113.4]**	86.5 [75.1, 100.2]	96.2 [82.9, 110.2]
Comp1	59.6 [56.4, 63.0]	71.6 [69.3, 73.8]	**105.7 [99.9, 111.4]**	97.2 [90.1, 104.7]	104.0 [97.9, 110.7]
Comp2	57.2 [53.3, 61.3]	77.5 [69.1, 86.4]	96.8 [92.9, 101.0]	91.1 [88.6, 93.8]	**98.9 [92.9, 105.2]**
Comp3	59.5 [52.7, 65.2]	77.4 [72.9, 84.2]	**109.7 [101.9, 116.4]**	86.0 [80.6, 91.4]	104.9 [98.4, 111.2]
Comp4	51.1 [44.6, 56.6]	66.9 [64.7, 69.0]	69.1 [64.4, 73.4]	**78.2 [73.8, 82.4]**	68.8 [62.7, 74.6]

Table 10: Table 7 with default hyperparameters (four parallel environments, 128 steps per PPO rollout, and all learning rates set to 10^{-3}).

Algorithm	Mean rank ↓	Beats MaxAgg-NoSI ↑	Beats MaxAgg-SI ↑
PPO	5.60 [5.33, 5.75]	0.57 [0.53, 0.70]	0.85 [0.78, 0.88]
MaxAgg-SI, $\lambda_{\mathrm{MA}} = 0.95$	7.70 [7.42, 7.79]	0.30 [0.23, 0.38]	—
MaxAgg-NoSI, $\lambda_{\mathrm{MA}} = 0.95$	6.83 [6.58, 7.01]	—	0.57 [0.50, 0.65]
SIO, $\xi = 0$	4.92 [4.60, 5.22]	0.65 [0.62, 0.75]	0.78 [0.72, 0.82]
SIO, $\xi = 0.5$	3.48 [3.25, 3.92]	**0.90 [0.82, 0.90]**	0.90 [0.82, 0.93]
AMSIO, $\xi = 0.5$	4.15 [3.75, 4.28]	0.82 [0.78, 0.88]	**0.97 [0.88, 0.97]**
ISSIO, $\xi = 0.5$	**3.35 [3.05, 3.62]**	**0.90 [0.85, 0.90]**	0.90 [0.85, 0.95]

Table 11: Table 8 with default hyperparameters (four parallel environments, 128 steps per PPO rollout, and all learning rates set to 10^{-3}).

Algorithm	Acrobot-v1	CartPole-v1	MountainCar-v0	PendulumDiscrete-v1
PPO	**108.9 [106.7, 110.7]**	119.8 [118.3, 121.2]	1.5 [0.0, 3.7]	**64.3 [56.5, 72.9]**
MaxAgg-SI, $\lambda_{\mathrm{MA}} = 0.95$	87.7 [82.6, 92.4]	117.0 [116.3, 117.8]	5.9 [3.0, 9.1]	30.8 [27.1, 34.6]
MaxAgg-NoSI, $\lambda_{\mathrm{MA}} = 0.95$	88.2 [86.8, 89.6]	113.5 [111.2, 115.5]	14.2 [9.2, 19.7]	62.3 [58.3, 65.8]
SIO, $\xi = 0$	85.3 [82.9, 87.6]	114.6 [112.5, 116.7]	68.2 [58.0, 78.9]	58.2 [55.1, 61.4]
SIO, $\xi = 0.5$	102.5 [101.4, 103.6]	122.4 [122.0, 122.7]	**74.3 [64.2, 84.6]**	61.7 [56.7, 66.4]
AMSIO, $\xi = 0.5$	106.7 [106.1, 107.3]	**123.1 [122.8, 123.3]**	55.7 [47.3, 64.3]	54.2 [49.8, 58.7]
ISSIO, $\xi = 0.5$	104.6 [103.7, 105.4]	122.3 [121.9, 122.7]	70.9 [61.4, 80.9]	58.8 [54.5, 63.1]

Early experiments for both MaxAggregation and SIO suggested an actor learning rate grid of $\{10^{-6}, 10^{-4}, 10^{-2}\}$ and a critic learning rate grid of $\{10^{-5}, 10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}\}$, which we use for all algorithms (including PPO) and all tables (unless otherwise specified). (However, the final results perhaps suggest that a better hyperparameter search space for the actor learning rate grid for both algorithms might be $\{10^{-5}, 10^{-3}, 10^{-1}\}$.)

In the learning-rate sweep plots, we abbreviate the actor learning rate as `alr`, and the critic learning rate as `clr`.

For all MaxAggregation and MaxAggregation-SI results, we use $\lambda_{\mathrm{MA}} = 0.95$, and (following (Cheng et al., 2020)) $N_{\mathrm{pretrain}} = 16$ (per expert). For all SIO, AMSIO, and ISSIO results other than the β ablation, we use $\beta = 1$. We likewise use $\xi = 0.5$ unless otherwise specified.

E.2 Expert constructions

`G` (great expert). For each environment, the great expert `G` is a DQN policy trained for 500k environment steps with the best learning rate (by peak returns) in $\{10^{-5}, 3 \times 10^{-4}, \ldots, 10^{-1}\}$, saving the weights at the peak return. This gave learning rates 3×10^{-4} for `Acrobot-v1` and `CartPole-v1`, 3×10^{-3} for `MountainCar-v0`, and 10^{-2} for `PendulumDiscrete-v1`. Apart from the learning rate, we use cleanRL's defaults (Huang et al., 2022; CleanRL contributors, 2026a). The saved final Q-network acts greedily by taking $\mathrm{argmax}_a\, Q(s, a)$.

`R` (random expert). The random expert takes a uniformly random action at each step.

Every expert in `Comp1`–`Comp4` uses the same `G` policy, but acts uniformly randomly outside its per-expert great region. Within each set, the great regions partition the state space. We describe them below; values are clamped to the stated ranges before binning.

`Comp1` (position split). A scalar position-related score is computed from the state, and one expert acts randomly where this score is ≥ 0, the other where it is < 0.

- `CartPole-v1`: pole angle.
- `MountainCar-v0`: car position $+0.5$ (so the split is at the bottom of the valley, $x = -0.5$).
- `Acrobot-v1`: end-effector horizontal coordinate, $\sin(\theta_1) + \sin(\theta_1 + \theta_2)$.
- `PendulumDiscrete-v1`: $\sin(\theta)$ (horizontal component of the pole tip).

`Comp2` (velocity split). Same ≥ 0 vs. < 0 convention as `Comp1`, but on a velocity coordinate of the state.

- `CartPole-v1`: pole angular velocity.
- `MountainCar-v0`: car velocity.
- `Acrobot-v1`: angular velocity of the second link, $\dot{\theta}_2$.
- `PendulumDiscrete-v1`: angular velocity, $\dot{\theta}$.

`Comp3` (2-class position checkerboard). One or two position-related features are uniformly binned over the listed range, and the parity (sum of bin indices, mod 2) determines which expert is great there. The two experts are great on opposite parities.

- `CartPole-v1`: 2-D, cart position over $[-2.4, 2.4]$ in 20 bins $\times$ pole angle over $[-\pi/15, \pi/15]$ in 20 bins.
- `MountainCar-v0`: 1-D, car position over $[-1.2, 0.6]$ in 101 bins.
- `Acrobot-v1`: 2-D, end-effector $x = \sin(\theta_1) + \sin(\theta_1 + \theta_2)$ over $[-2, 2]$ in 20 bins $\times$ end-effector $y = -\cos(\theta_1) - \cos(\theta_1 + \theta_2)$ over $[-2, 2]$ in 20 bins.
- `PendulumDiscrete-v1`: 1-D, $\sin(\theta)$ over $[-1, 1]$ in 101 bins.

`Comp4` (8-class position checkerboard). Same construction as `Comp3`, but the bin-index sum is taken mod 8, and bin counts are slightly larger to make them divisible by 8 (in retrospect, ideally we

would have varied only the number of classes compared to the 2-class checkerboard, but the number of bins remains close to the 2-class checkerboard). Expert $i \in \{1, \ldots, 8\}$ is great only on class $i - 1$ and acts randomly elsewhere.

- `CartPole-v1`: cart position in 24 bins $\times$ pole angle in 24 bins (same ranges as `Comp3`).
- `MountainCar-v0`: car position over $[-1.2, 0.6]$ in 104 bins.
- `Acrobot-v1`: end-effector x in 24 bins $\times$ end-effector y in 24 bins (same ranges as `Comp3`).
- `PendulumDiscrete-v1`: $\sin(\theta)$ over $[-1, 1]$ in 104 bins.

F Learning-rate sweep plots

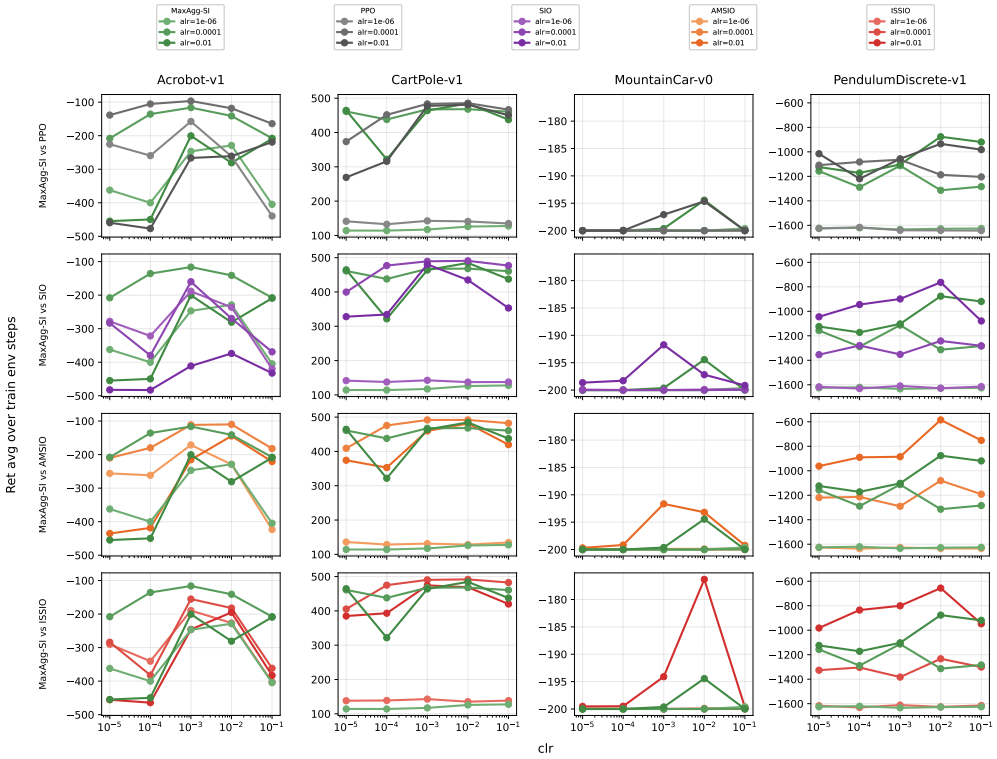

Figure 3: Mean evaluation returns over training for expert set R, with MaxAggregation-SI.

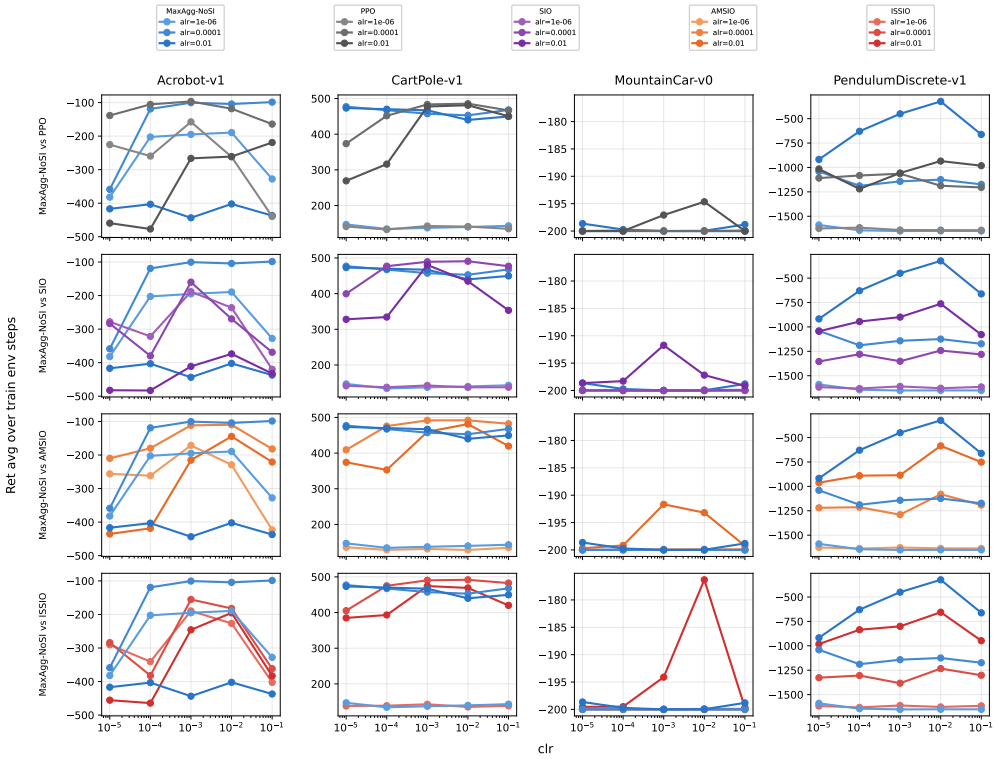

Figure 4: Mean evaluation returns over training for expert set R, with MaxAggregation-NoSI.

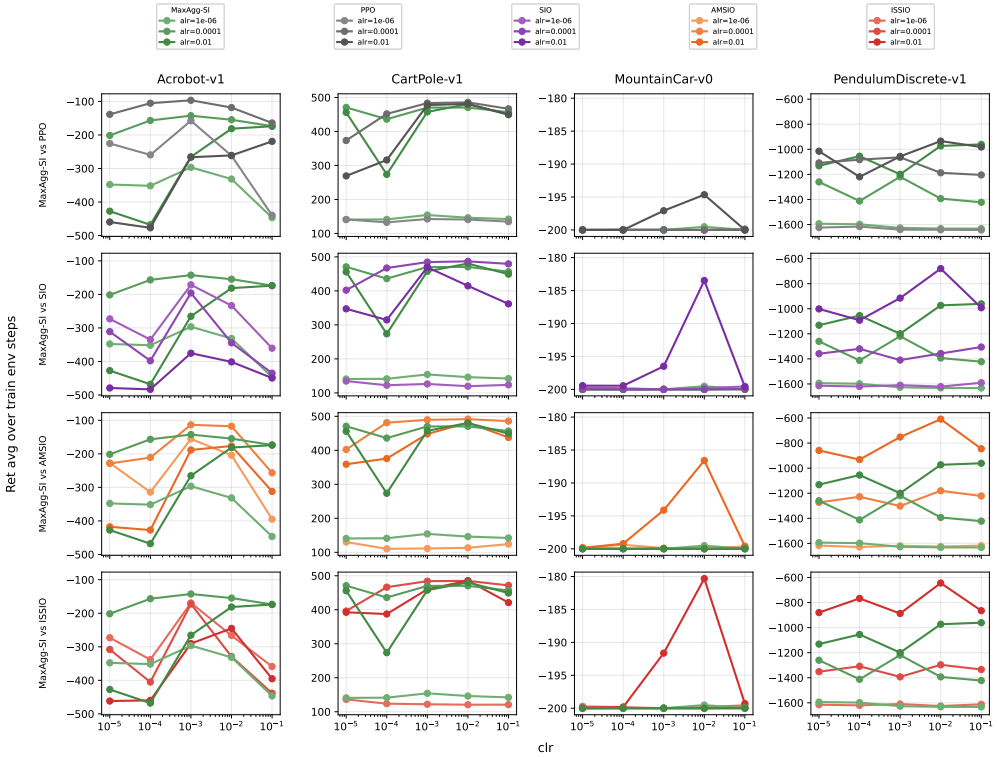

Figure 5: Mean evaluation returns over training for expert set 2R, with MaxAggregation-SI.

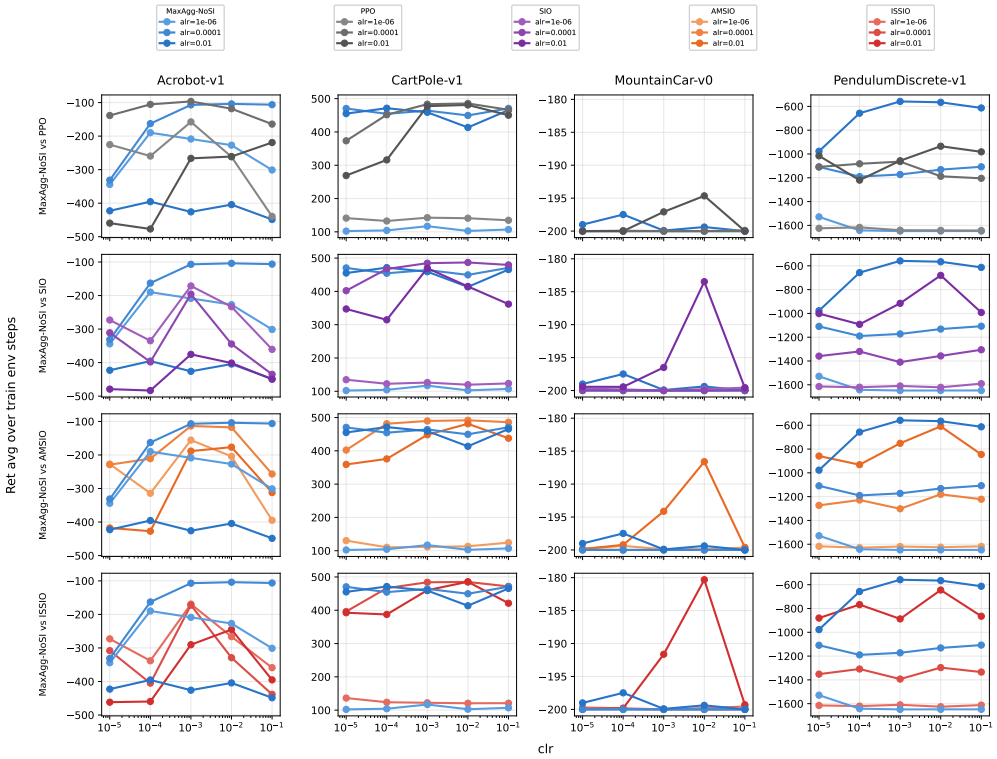

Figure 6: Mean evaluation returns over training for expert set 2R, with MaxAggregation-NoSI.

Figure 7: Mean evaluation returns over training for expert set 8R, with MaxAggregation-SI.

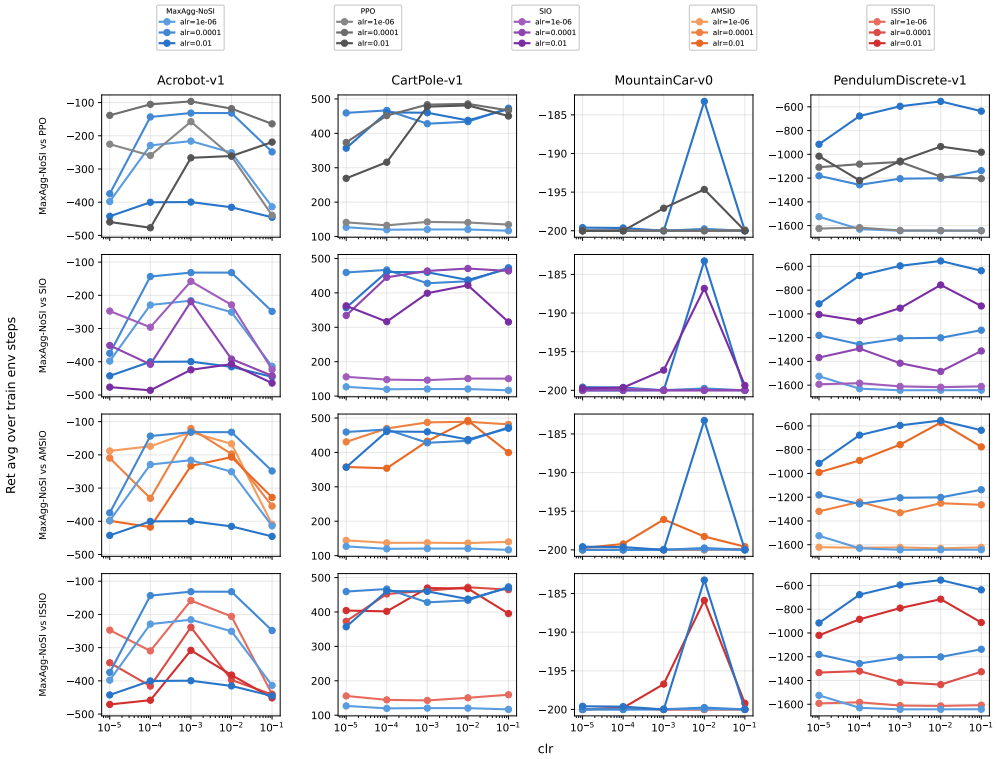

Figure 8: Mean evaluation returns over training for expert set 8R, with MaxAggregation-NoSI.

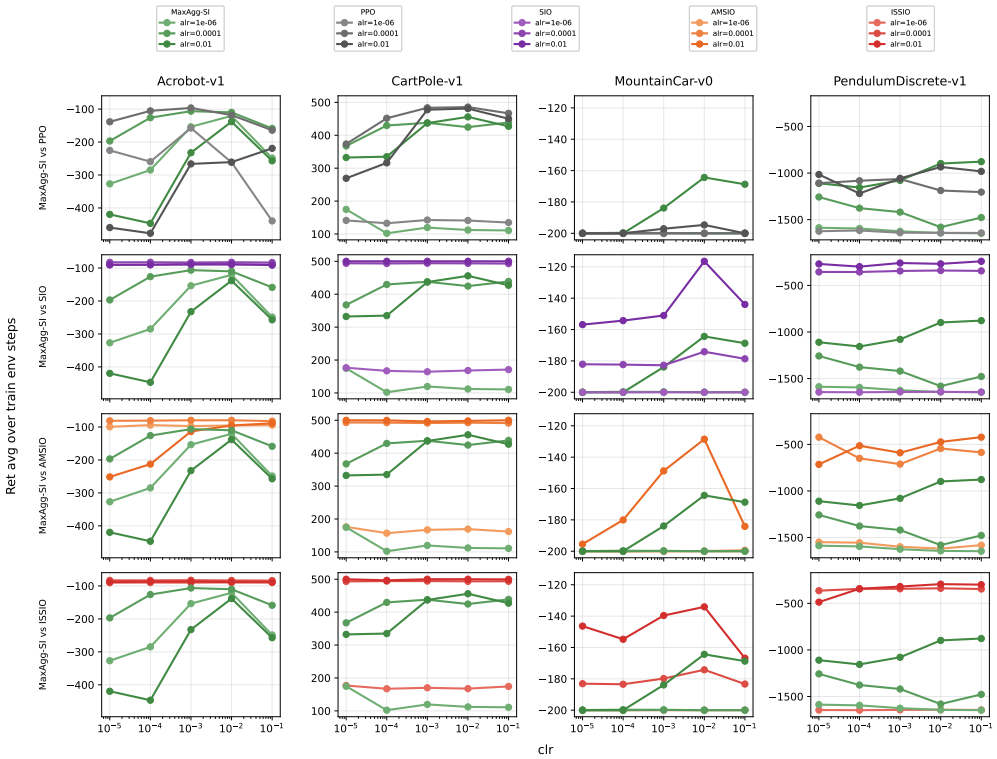

Figure 9: Mean evaluation returns over training for expert set G, with MaxAggregation-SI.

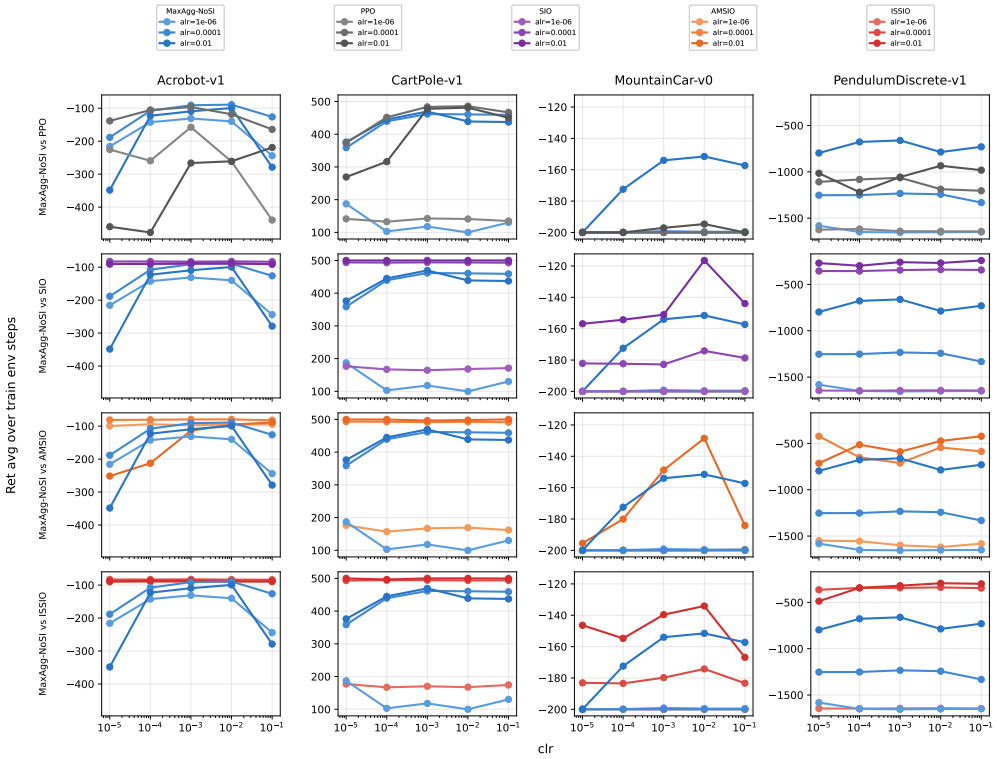

Figure 10: Mean evaluation returns over training for expert set G, with MaxAggregation-NoSI.

Figure 11: Mean evaluation returns over training for expert set G2R, with MaxAggregation-SI.

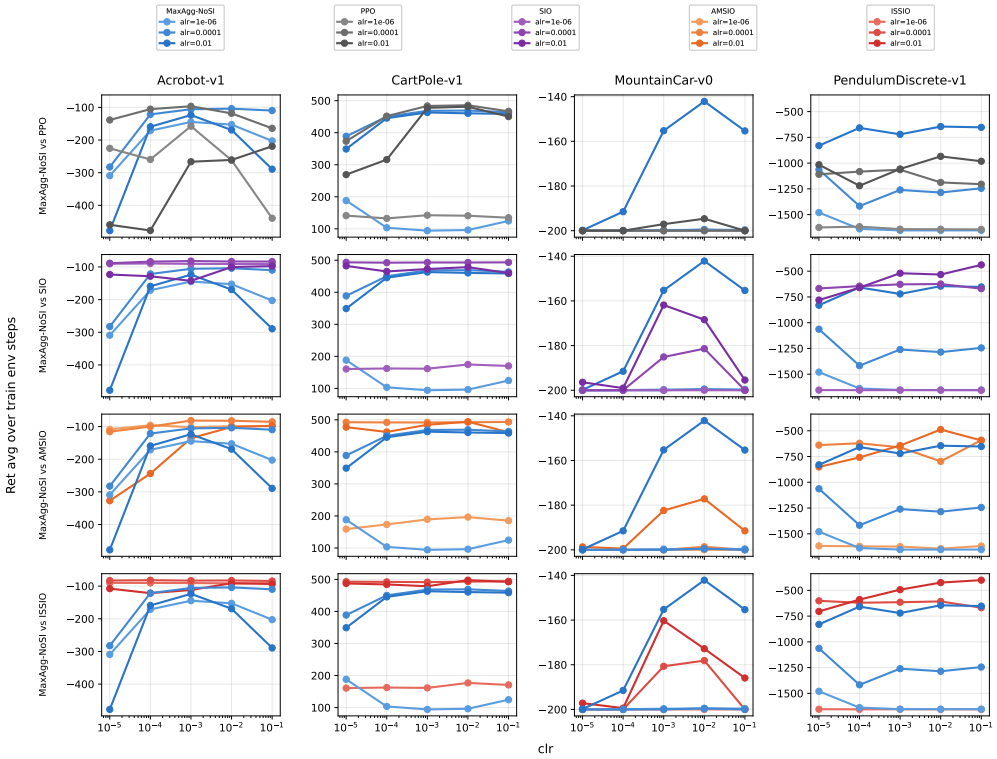

Figure 12: Mean evaluation returns over training for expert set G2R, with MaxAggregation-NoSI.

Figure 13: Mean evaluation returns over training for expert set G7R, with MaxAggregation-SI.

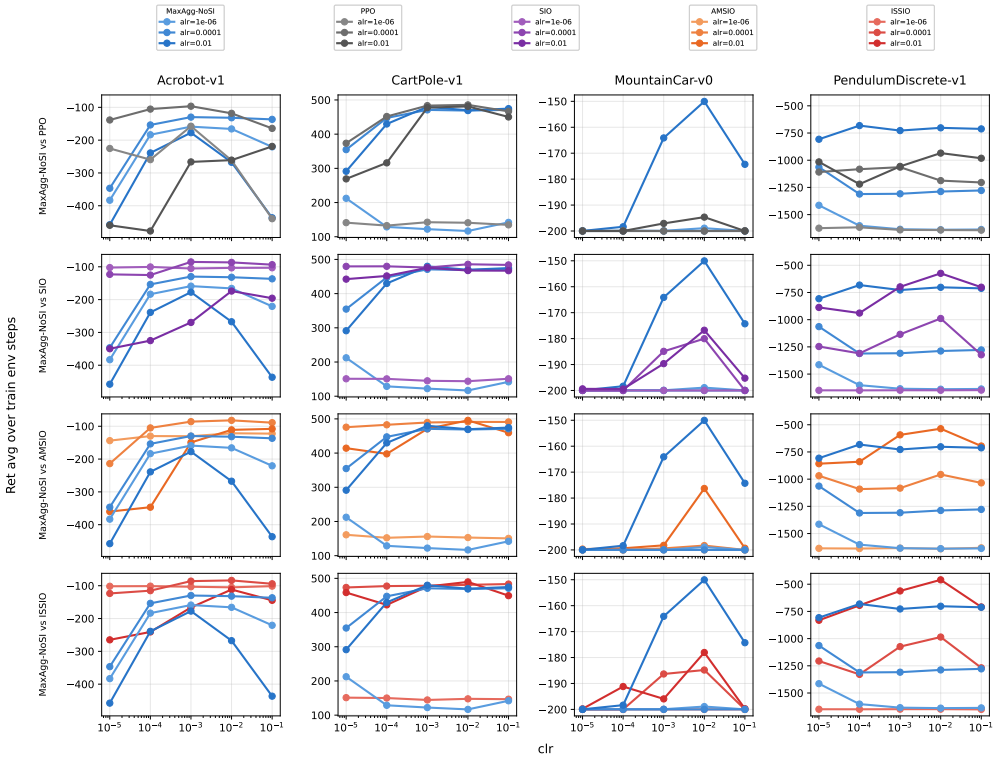

Figure 14: Mean evaluation returns over training for expert set G7R, with MaxAggregation-NoSI.

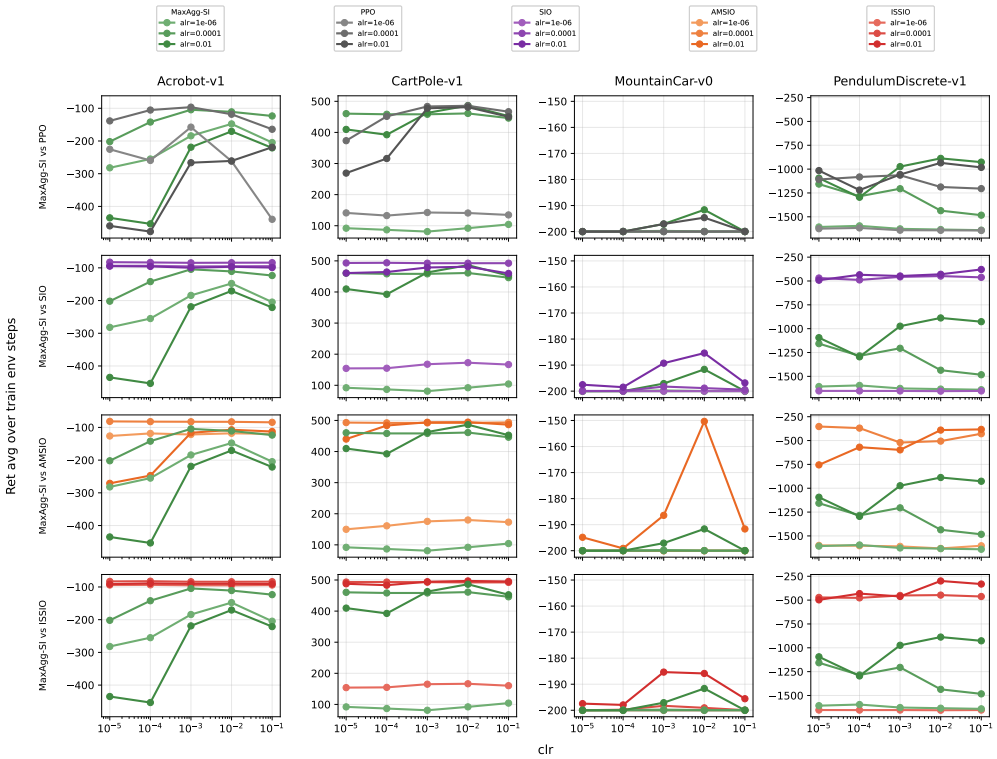

Figure 15: Mean evaluation returns over training for expert set `Comp1`, with MaxAggregation-SI.

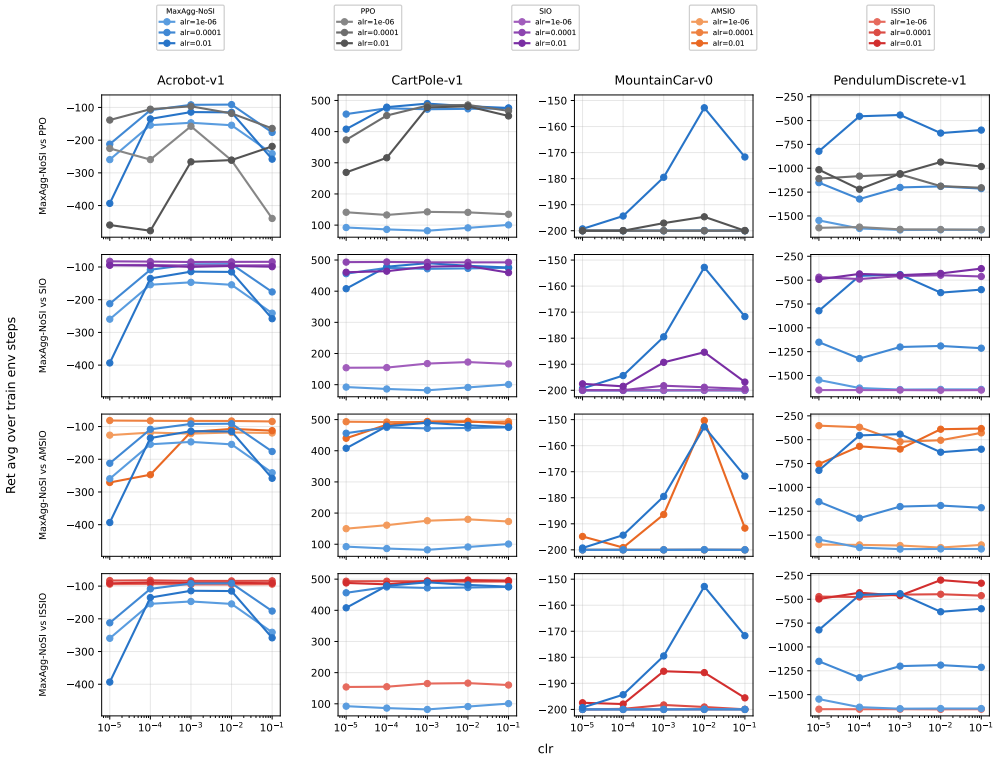

Figure 16: Mean evaluation returns over training for expert set `Comp1`, with MaxAggregation-NoSI.

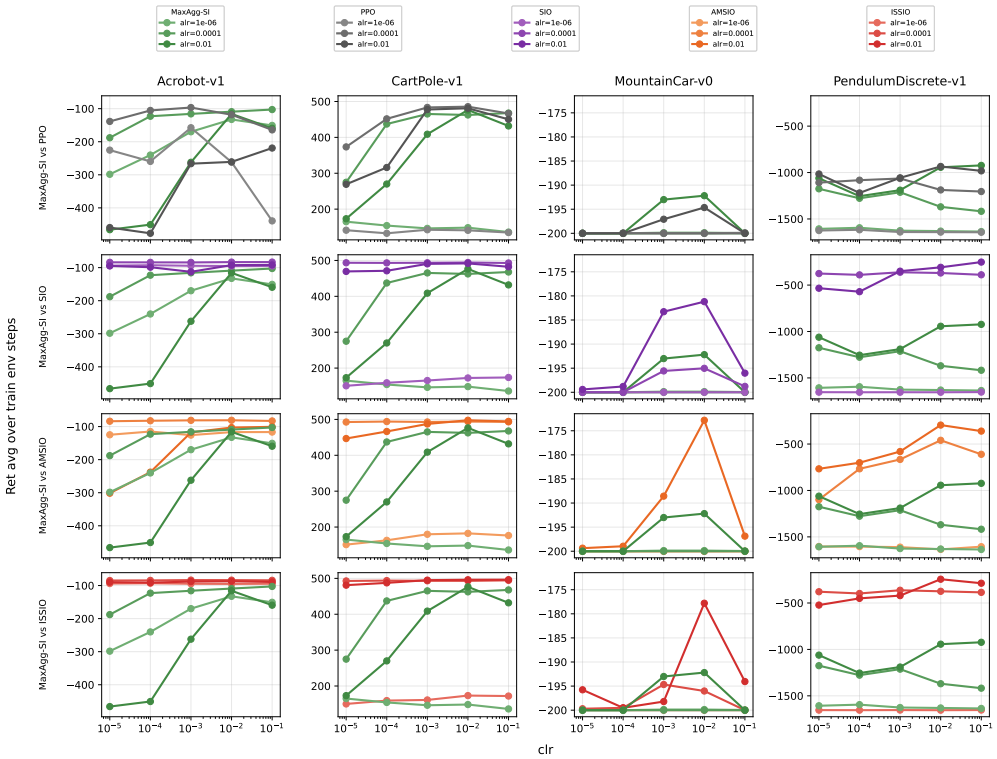

Figure 17: Mean evaluation returns over training for expert set `Comp2`, with MaxAggregation-SI.

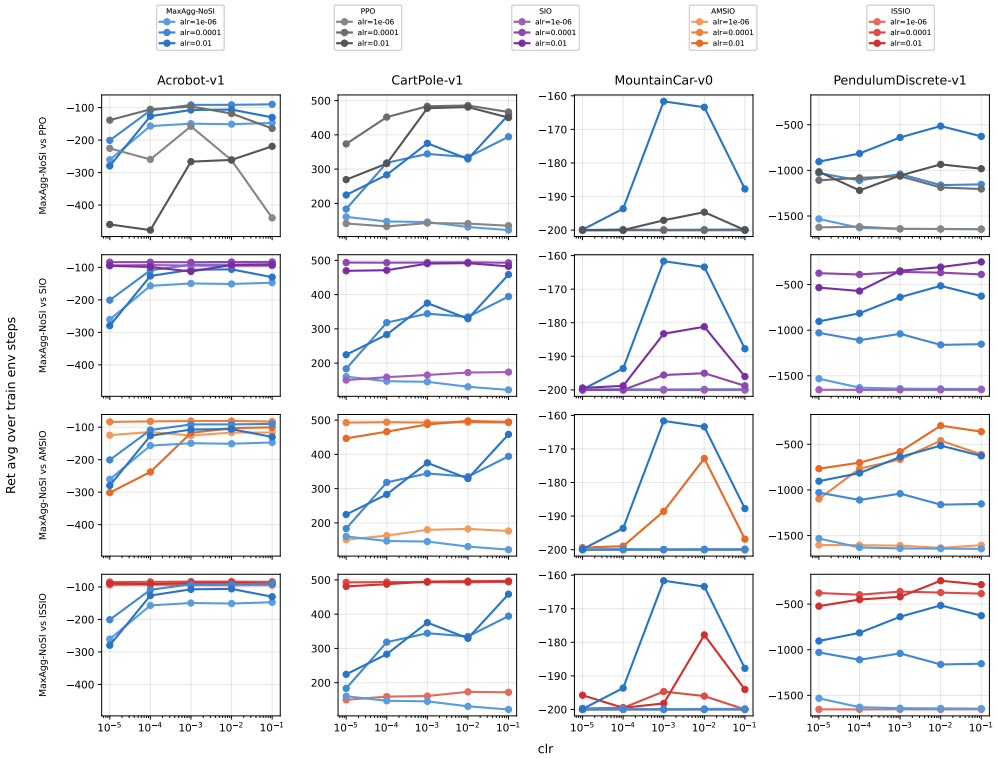

Figure 18: Mean evaluation returns over training for expert set `Comp2`, with MaxAggregation-NoSI.

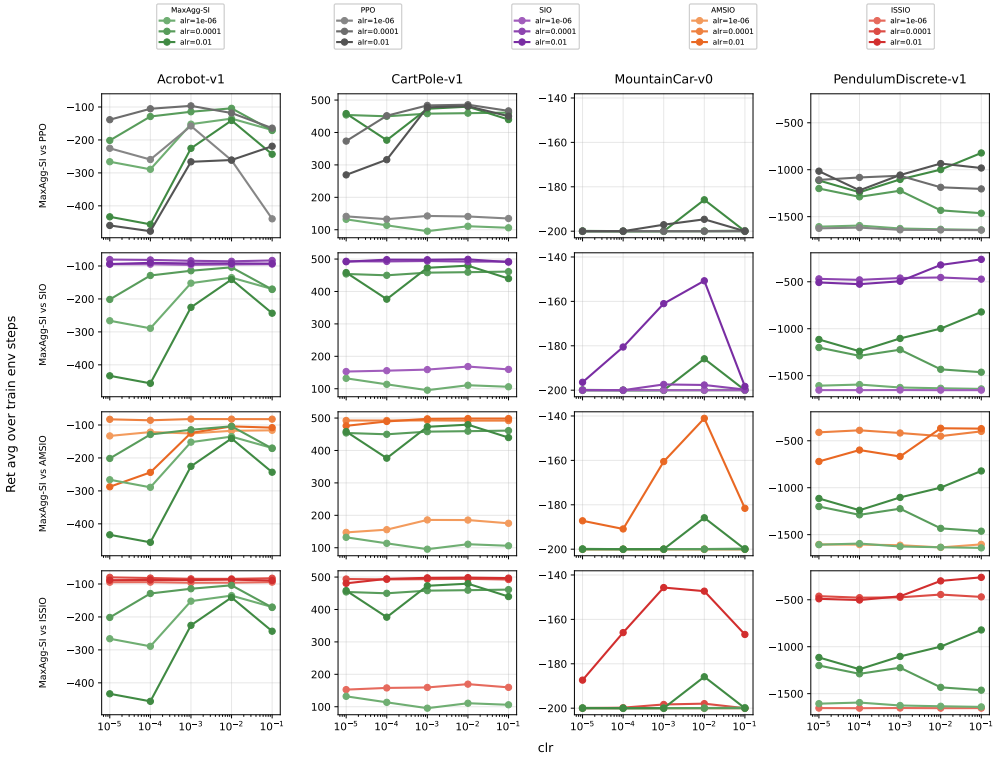

Figure 19: Mean evaluation returns over training for expert set `Comp3`, with MaxAggregation-SI.

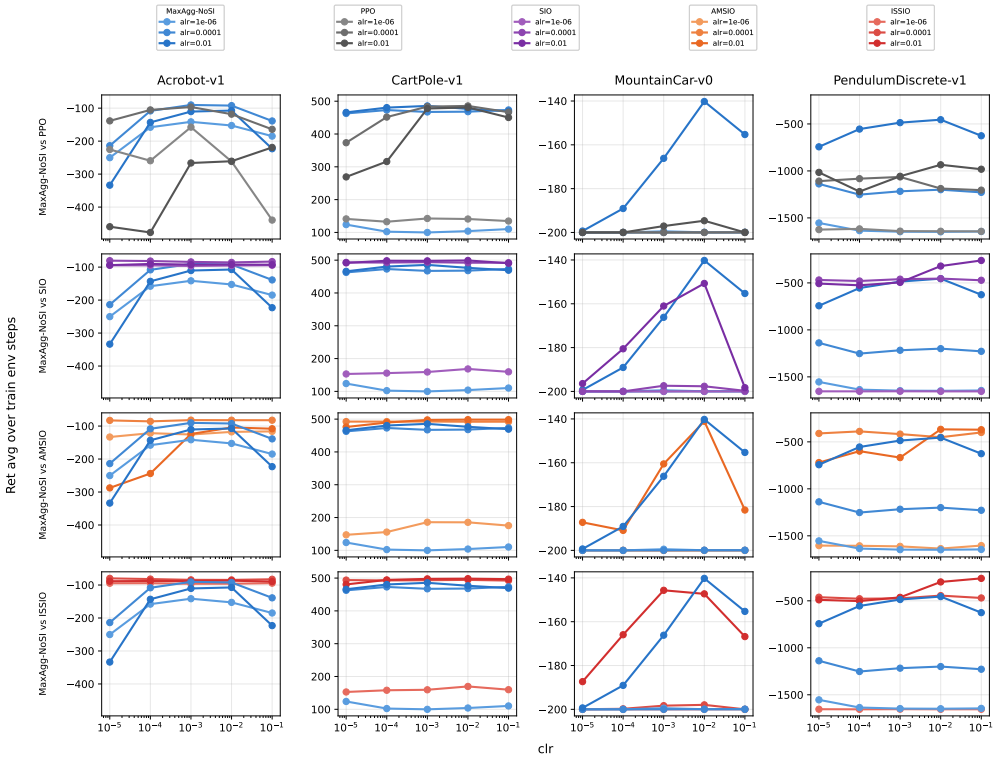

Figure 20: Mean evaluation returns over training for expert set `Comp3`, with MaxAggregation-NoSI.

Figure 21: Mean evaluation returns over training for expert set `Comp4`, with MaxAggregation-SI.

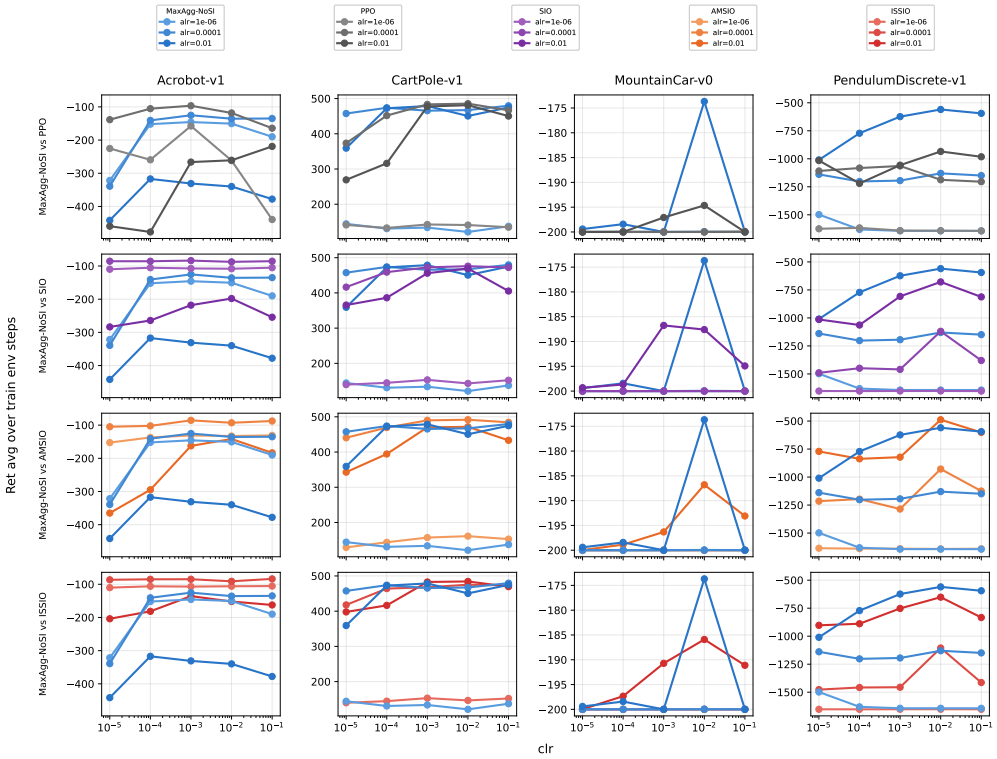

Figure 22: Mean evaluation returns over training for expert set `Comp4`, with MaxAggregation-NoSI.

G Expert-selection plots

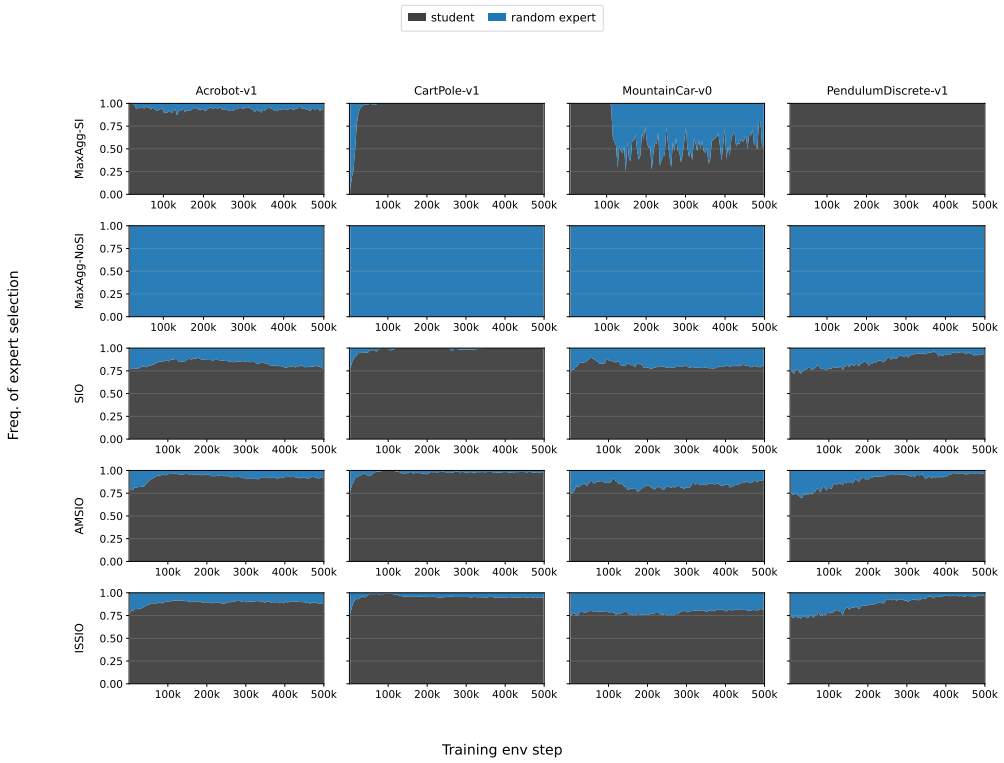

Figure 23: Expert-selection frequencies for expert set R. All learning rates at 10^{-3}.

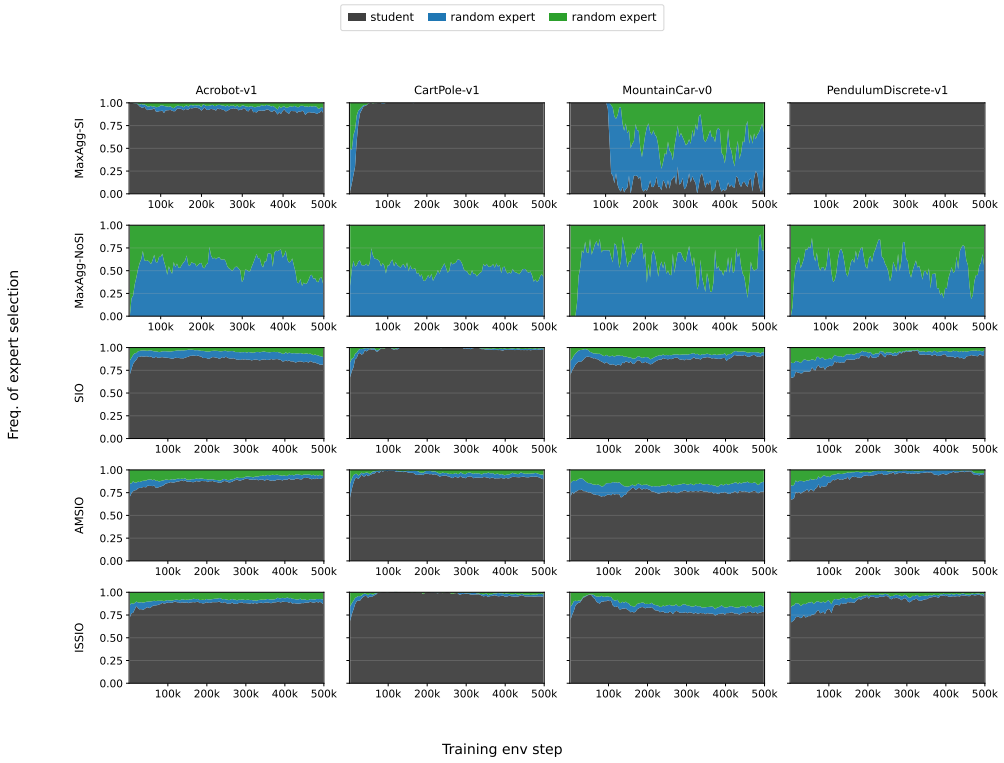

Figure 24: Expert-selection frequencies for expert set 2R. All learning rates at 10^{-3}.

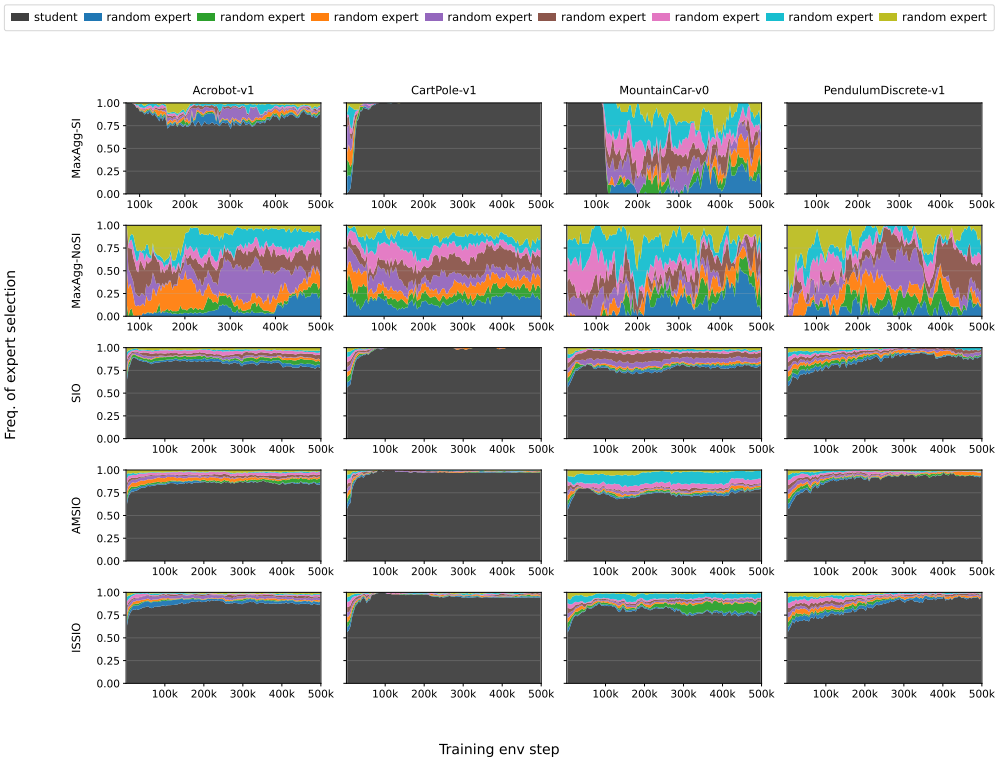

Figure 25: Expert-selection frequencies for expert set 8R. All learning rates at 10^{-3}.

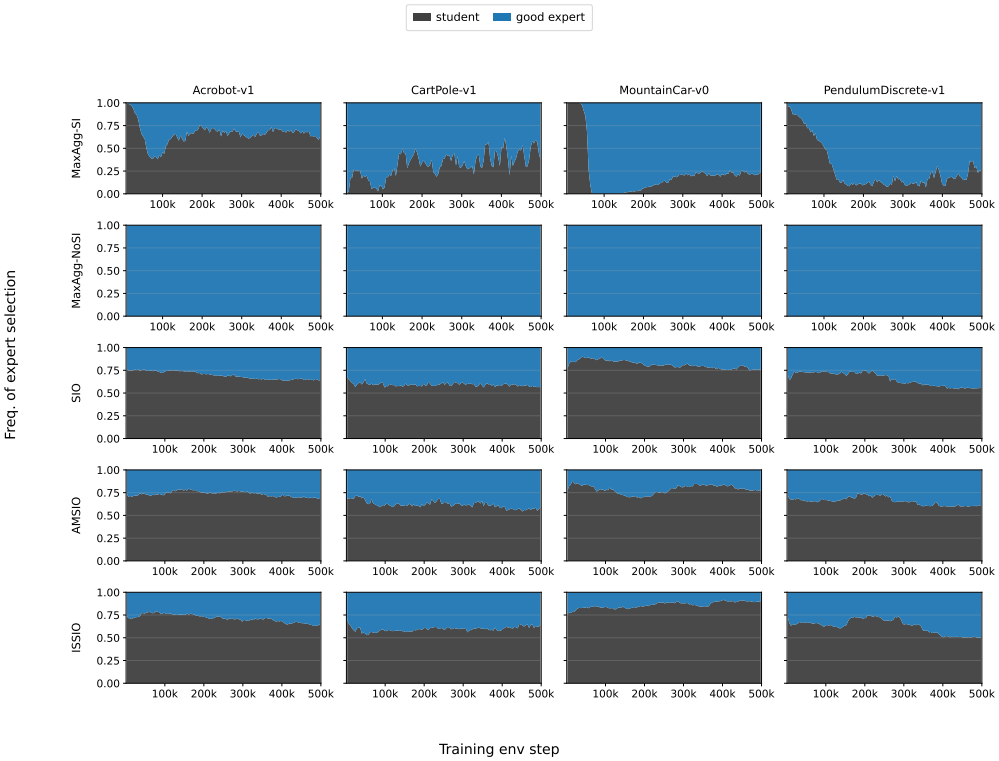

Figure 26: Expert-selection frequencies for expert set G. All learning rates at 10^{-3}.

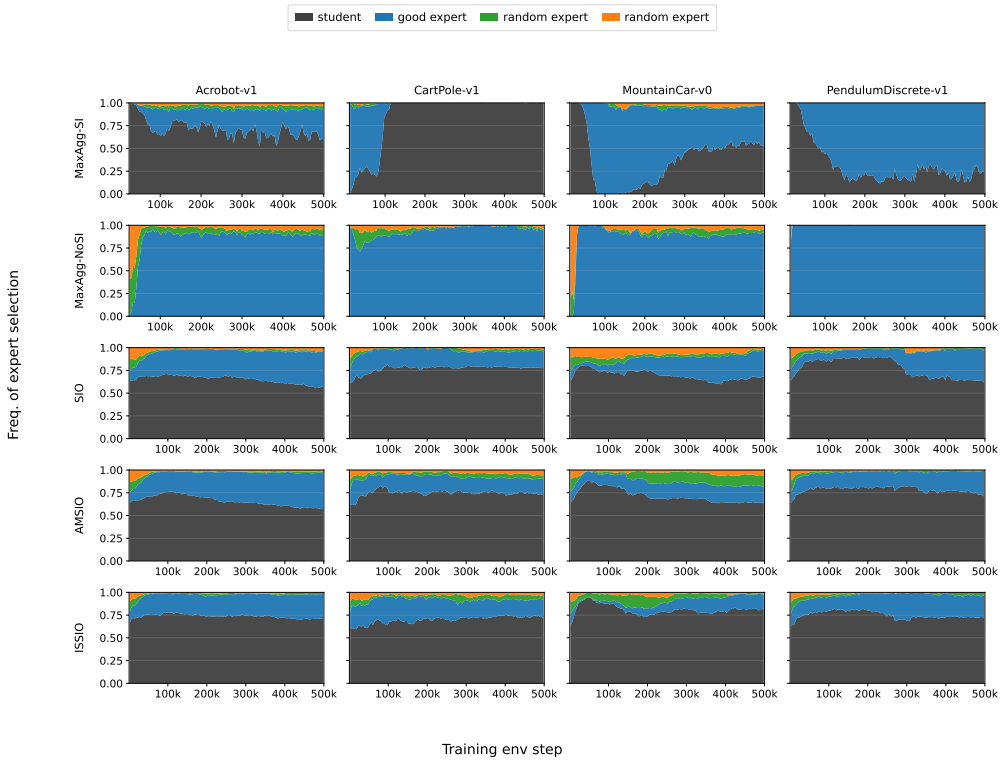

Figure 27: Expert-selection frequencies for expert set G2R. All learning rates at 10^{-3}.

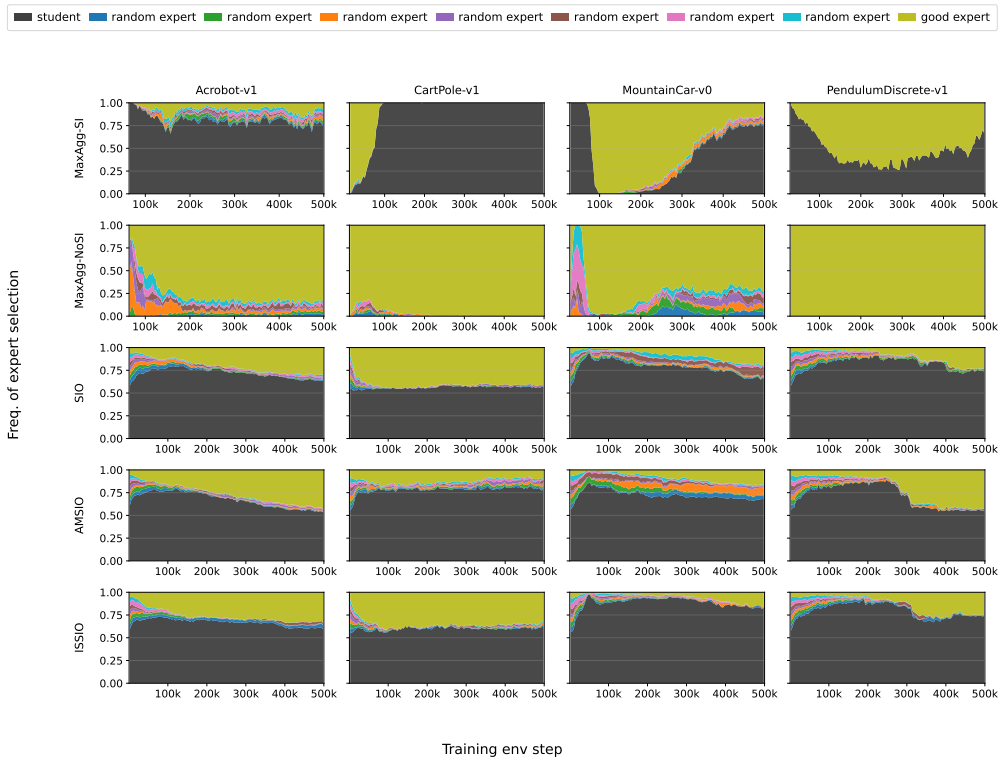

Figure 28: Expert-selection frequencies for expert set G7R. All learning rates at 10^{-3}.

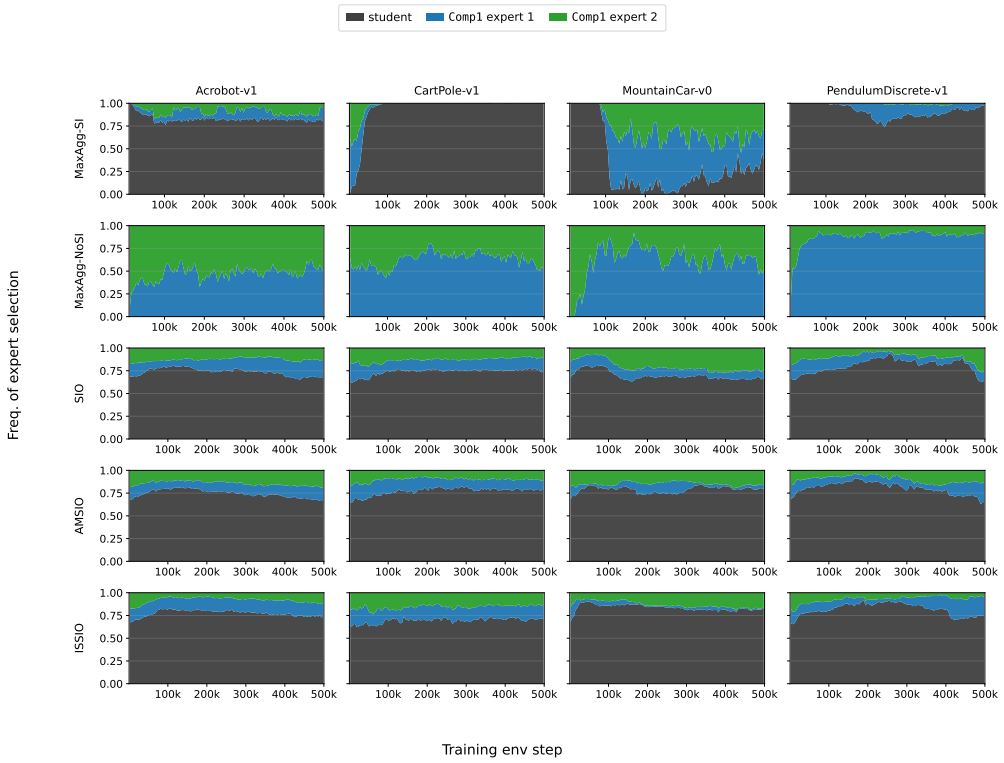

Figure 29: Expert-selection frequencies for expert set `Comp1`. All learning rates at 10^{-3}.

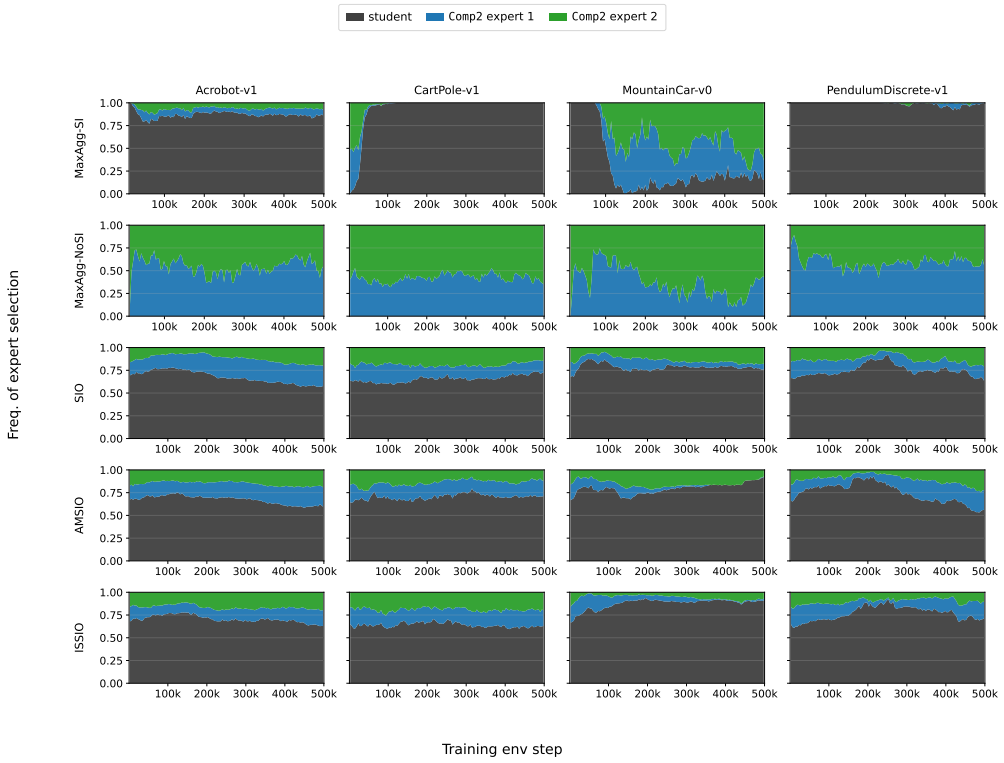

Figure 30: Expert-selection frequencies for expert set `Comp2`. All learning rates at 10^{-3}.

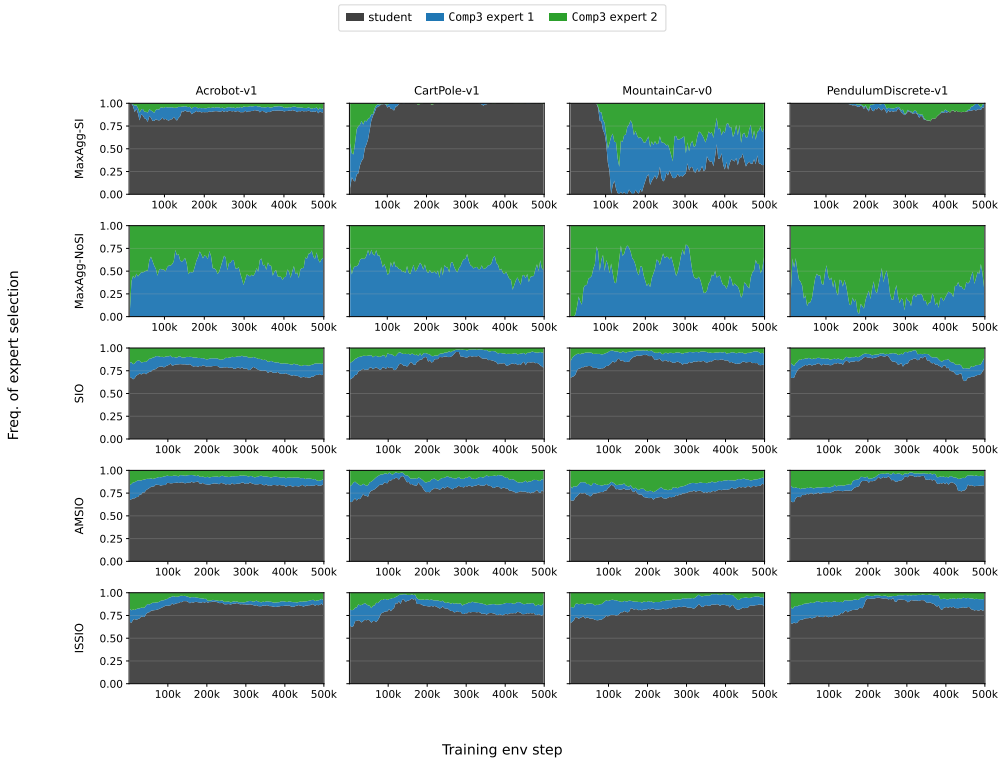

Figure 31: Expert-selection frequencies for expert set `Comp3`. All learning rates at 10^{-3}.

Figure 32: Expert-selection frequencies for expert set `Comp4`. All learning rates at 10^{-3}.

H Additional Reproducibility Details

H.1 Compute Resources

All experiments were run as CPU jobs on an internal SLURM cluster. No GPU acceleration was used. Each individual training run used one SLURM task with two CPU cores and 4 GB of RAM.

For a single seed of a single {algorithm, environment, expert set, actor learning rate, critic learning rate} configuration, we allocated 2 CPU cores, 4 GB RAM, for roughly 10 minutes of runtime. The main table sweeps five expert-using algorithms over 10 expert sets, 4 environments, 3 actor learning rates, 5 critic learning rates, and 10 random seeds: $5 \times 10 \times 4 \times 3 \times 5 \times 10 = 30{,}000$ individual CPU jobs, or roughly $30{,}000 \times 2 \times (10/60) = 10{,}000$ CPU-core-hours. The PPO comparison under the same learning-rate grid adds $4 \times 3 \times 5 \times 10 = 600$ jobs, or roughly 200 CPU-core-hours.

The appendix ablations and fixed-learning-rate figures use the same per-run resources. Counting the additional SIO variants reported in Tables 2 to 4 gives roughly another 24,000 grid-search jobs, and the fixed-learning-rate tables and learning curves add a few thousand more jobs. Thus, the reported experimental results required roughly 6×10^4 individual CPU jobs, or roughly 2×10^4 CPU-core-hours.

The full research project used more compute than the reported runs. In particular, preliminary sweeps were used to debug algorithm variants and to choose the learning-rate grid in Section E.1. Those early experiments used far less compute than the main experiments.

H.2 Experimental statistical significance

We omit error bars on learning rate sweep plots to keep the many lines on the plots readable.

H.3 Broader impacts

Our work has no societal impact beyond the generic societal impact of improved reinforcement learning + imitation learning algorithms.