Imitating the Imperfect: Offline-to-Online Robust Imitation Learning from Heterogeneous Demonstrators

Cheng Pan [*1] **Nan Lu** [*2] **Kai Arulkumaran** [3] **Josie Hughes** [1]

Abstract

Imitation learning (IL) typically relies on large-scale *offline* demonstrations collected from multiple human or algorithmic demonstrators. However, most existing approaches assume these demonstrators are homogeneous or near-optimal experts, a convenient but unrealistic assumption in real-world applications. In practice, demonstrations are often collected from heterogeneous demonstrators with unknown and varying levels of expertise, resulting in highly mixed-quality data. In this work, we study a challenging yet practical setting of imitation learning from such heterogeneous imperfect demonstrators. We propose *Latent Expertise and Optimality Scoring based Imitation Learning* (LEOS-IL), a principled framework that jointly learns (i) a state-action optimality scoring model and (ii) latent expertise levels for each demonstrator from unlabeled suboptimal *offline* demonstrations. The learned scoring model is then integrated into an *online* policy optimization procedure, where the agent is trained to maximize estimated optimality scores while its online rollouts are iteratively leveraged to refine the scoring model, thereby mitigating distributional shift during online adaptation. We further provide theoretical guarantees for optimal policy recovery and convergence of the proposed method. Experiments on continuous control benchmarks demonstrate that our approach consistently outperforms existing baselines across highly heterogeneous and low-quality demonstration regimes.

[*]Equal contribution [1]EPFL, Lausanne, Switzerland [2]University of Bristol, Bristol, UK [3]Araya Inc., Tokyo, Japan. Correspondence to: Cheng Pan <cheng.pan@epfl.ch>, Nan Lu <nan.lu@bristol.ac.uk>.

Accepted at ICML 2026 Workshop on Decision-Making from Offline Datasets to Online Adaptation: Black-Box Optimization to Reinforcement Learning. Copyright 2026 by the author(s).

1. Introduction

Imitation learning (IL) offers a convenient framework that enables agents to acquire skills directly from expert demonstrations (Zare et al., 2024). Prior research has shown that effective policies can be learned efficiently when provided with large-scale and high-quality demonstrations (Belkhale et al., 2023; Saxena et al., 2025). However, in real-world scenarios, large-scale datasets are often collected from *multiple* demonstrators, and human or algorithmic demonstrators typically possess strengths in specific contexts and may perform suboptimally outside their areas of expertise. For example, in autonomous driving data collected from electric vehicles (Lee et al., 2022; Zhang et al., 2024), drivers demonstrate a wide range of skill levels, resulting in *heterogene*ous and *imperfect* trajectories with unknown degrees of expertise. Treating all such demonstrators as perfect experts can lead to unreliable or even unsafe agent behavior, posing serious risks in safety-critical applications.

In this work, we study IL from heterogeneous imperfect demonstrators whose expertise levels are unknown and variable, naturally leading to datasets containing suboptimal demonstrations, and learn a policy online based on the estimated demonstrator expertise and data optimality. Existing methods (see details in Section 2) addressing this setting often rely on one or more of the following assumptions: (i) demonstrators are homogeneous, i.e., with the same expertise levels, and the optimal data dominates the dataset, (ii) access to explicit labels indicating the optimality of demonstrations, or (iii) the expertise level of each demonstrator is known a priori. These requirements are demanding and often impractical when demonstrations come from uncontrolled or large-scale sources.

This brings us to a central question: *Can we design effective IL algorithms without relying on these assumptions?* This problem setting is inherently challenging because neither demonstrator expertise levels nor demonstration optimality are observed. In the worst case, all demonstrators may exhibit very low expertise, making it extremely difficult to extract reliable signals for policy learning.

To address these challenges, we propose a novel *Latent Expertise and Optimality Scoring based Imitation Learn-*

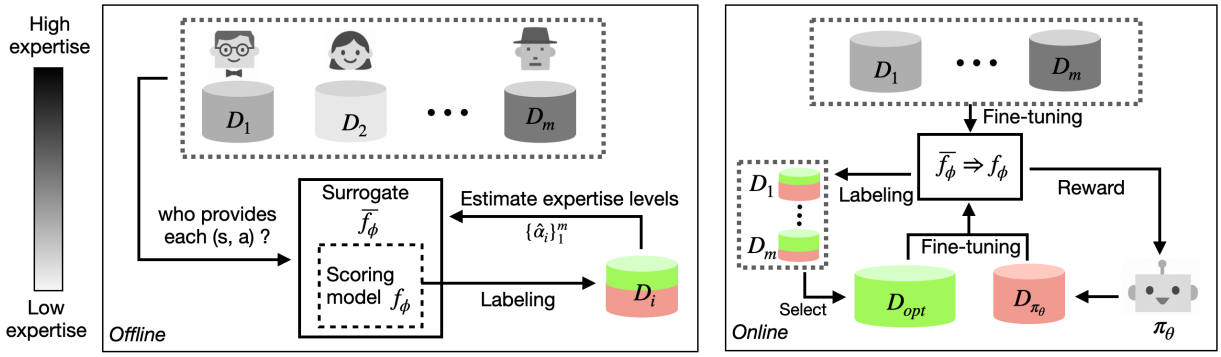


Figure 1. **Overview of our heterogeneous IL framework**, illustrated with m demonstrators possessing (unknown) heterogeneous expertise levels (represented in grayscale; darker shades indicate higher expertise). Green/red indicate inferred optimal/suboptimal pairs. **Stage 1** (left panel): Joint learning of *demonstrator expertise levels* $\{\alpha_i\}_{i=1}^m$ and the *optimality scoring model* f_ϕ through a a closed-loop process: (1) a surrogate classifier $\overline{f}_\phi$ is trained to predict the demonstrator identity for each (s, a) pair, which induces f_ϕ; (2) $\{\hat{\alpha}_i\}_{i=1}^m$ are updated based on predictions from f_ϕ. **Stage 2** (right panel): Iterative refinement of both f_ϕ and the agent policy π_θ using the demonstrator dataset $D = \{(i, D_i)\}_{i=1}^m$, a selected set of optimal demonstrations D_{opt}, and agent-generated samples D_{π_θ}.

ing framework (LEOS-IL) for learning from heterogeneous imperfect demonstrators. Our main contributions are summarized as follows.

Contribution 1: A formulation for modeling heterogeneous imperfect demonstrators. We introduce a principled formulation in which each demonstrator is modeled as a flexible mixture of optimal and suboptimal behaviors. We characterize a demonstrator's *expertise level* as the probability of selecting actions aligned with an underlying latent optimal policy. These expertise levels vary across demonstrators, naturally capturing heterogeneous demonstration quality, while remaining fully latent and unobserved. This formulation provides a principled foundation for learning from realistic mixed-quality datasets without requiring explicit supervision or curated expert data.

Contribution 2: A principled offline-to-online heterogeneous IL framework. Building on this formulation, we propose a novel two-stage IL framework (Figure 1). In the offline stage (Stage 1), we jointly estimate demonstrator expertise and learn a *scoring model* that assigns *optimality scores* to each state-action pair, representing the likelihood that the pair corresponds to the (latent) optimal policy. This is achieved through an expectation–maximization-style (EM-style) iterative procedure, which we prove converges under mild assumptions. In the online stage (Stage 2), the learned scoring model is used as a surrogate reward function for policy optimization. To address the distributional shift arising from offline training, we introduce an online iterative refinement procedure that alternates between policy learning and scoring model fine-tuning, yielding a progressively less biased and more informative reward signal.

Contribution 3: Theoretical and empirical validation. We provide theoretical guarantees for both optimal policy recovery and convergence of the proposed latent expertise estimation procedure. Empirically, we evaluate our method in two challenging heterogeneous scenarios: (i) a *wide-range setting*, with demonstrator expertise levels spanning $[0.1, 0.9]$, and (ii) a *low-quality setting*, where all demonstrators are highly suboptimal with expertise in $[0.05, 0.15]$. Across both regimes, our method outperforms baselines and, remarkably, achieves performance comparable to methods trained on the subset of purely optimal demonstrations. These results highlight the practical value of our framework in realistic IL settings where demonstrator quality is unknown, uncurated, and highly variable.

2. Related work

IL with Homogeneous Demonstrators A large body of IL research assumes that demonstrations originate from a single, near-optimal expert policy. This assumption underpins classical approaches such as *behavior cloning* (BC; Pomerleau, 1988), as well as more recent methods such as *generative adversarial imitation learning* (GAIL; Ho & Ermon, 2016), which matches the state-action occupancy measures of the learner and the demonstrator via adversarial training. Methods designed to handle noisy demonstrations, such as *robust imitation learning* (RIL; Tangkaratt et al., 2020), also implicitly retain this homogeneity assumption by modeling demonstrator behavior as arising from a mixture of optimal and suboptimal policies, with a single mixing coefficient representing the expertise level, and by assuming that expert data dominates the dataset. While effective under relatively homogeneous conditions, these methods may be less well suited to settings where demonstrator expertise varies widely.

Table 1. Comparison of IL methods. ✓ = satisfies criterion, ✗ = does not. "Non-dominant optimal demo" means the method does not require optimal demonstrations to dominate the dataset. "Multi-demo support" refers to the ability to handle demonstrations from multiple demonstrators with varying expertise levels. "Handles structured suboptimality" indicates whether the method can manage structured (i.e., non-random) suboptimal demonstrations. "No optimality labels needed" indicates no labels about the optimality of demonstrations is needed. "Probabilistic demo scoring" denotes whether the method infers a probabilistic measure of demonstration quality. "Interactive IL" denotes whether the method can improve with online environment interaction (without the ground truth reward function).

Method	BC	GAIL	RIL	IC-GAIL	UID-GAIL	RCE	WGAIL	ILEED	Ours
Non-dominant optimal demo	✗	✗	✗	✓	✓	✗	✗	✓	✓
Multi-demo support	✗	✗	✗	✗	✗	✗	✓	✓	✓
Handles structured suboptimality	✗	✗	✓	✓	✓	✗	✓	✗	✓
No optimality labels needed	✗	✗	✓	✗	✓	✗	✓	✓	✓
Probabilistic demo scoring	✗	✓	✓	✗	✓	✓	✗	✗	✓
Interactive IL	✗	✓	✓	✓	✓	✓	✓	✗	✓

IL with Optimality Labels To better handle suboptimal demonstrations, many IL methods use supervision to distinguish optimal from non-optimal behavior. Extensions of GAIL, such as *imperfect demonstration and confidence GAIL* (IC-GAIL; Wu et al., 2019) uses confidence labels to reweight imperfect demonstrations, whereas *positive-unlabeled-style GAIL for unlabeled imperfect demonstrations* (UID-GAIL; Wang et al., 2023) does not assume labeled successful trajectories. Instead, it treats the demonstration set as an unlabeled mixture and uses a positive-unlabeled-style risk decomposition based on the current policy occupancy to guide learning toward the residual demonstration distribution. Similarly, example-based methods like *recursive classification of examples* (RCE; Eysenbach et al., 2021) train success classifiers from labeled examples to create proxy rewards. These methods are effective when labeled data are available, but their applicability can be limited when such supervision is difficult to obtain.

IL without Homogeneity or Labels More recent efforts aim to learn from heterogeneous, unlabeled, and suboptimal demonstrations without the above assumptions. *Weighted GAIL* (WGAIL; Wang et al., 2021) estimates importance weights from discriminator outputs, emphasizing near-optimal samples without prior knowledge of expertise. *Imitation learning by estimating expertise of demonstrators* (ILEED; Beliaev et al., 2022) jointly learns a policy and state-dependent expertise via offline maximum-likelihood estimation, but assumes suboptimality is uniform action selection, making it sensitive to limited state coverage and structured suboptimality. Furthermore, ILEED does not allow improvement from environment interaction. In contrast, we model expertise at two levels: a demonstrator-level expertise capturing overall skill, and a state–action optimality score capturing context-specific expertise. We learn a surrogate scoring function that predicts state–action expertise and serves as a surrogate reward for online reinforcement learning, enabling robust estimation without full state coverage and effective learning from structured, heterogeneous, and low-quality demonstrations.[1]

3. IL from heterogeneous demonstrators

In this section, we formally formulate the problem of IL from heterogeneous demonstrators and propose our method.

3.1. Problem Setup

In reinforcement learning (RL; Sutton & Barto, 1998), the optimal policy maximizes the expected cumulative reward: $\pi^* = \text{argmax}_\pi \mathbb{E}_{p_\pi}\left[\sum_{t=1}^{T} \gamma^t r(s_t, a_t)\right]$, where $r(s_t, a_t)$ is the reward associated with the state-action pair (s, a) at time t, p_π is the trajectory distribution induced by policy π, and γ is the discount factor. Unlike RL, in IL the reward function is unobserved and learning relies on a dataset of demonstrator trajectories $\tau = \{(s_0, a_0), ..., (s_{T_\tau}, a_{T_\tau})\}$ of varying lengths T_τ. The goal is to learn a policy π that matches the optimal policy's state-action distribution.

To make this feasible, most IL algorithms assume that *all* demonstrations are generated by the optimal policy. However, this assumption is restrictive as real datasets are often collected from multiple demonstrators with varying skills. Therefore, in this work, we study IL from heterogeneous demonstrators with different and unknown expertise.

Dataset: We consider m demonstrators. Each demonstrator $i \in [m]$ provides a set of trajectories D_i and is associated with an *expertise level* $\alpha_i \in [0, 1]$. We define α_i as the probability that any given (s, a) pair from D_i was generated by the optimal policy, i.e., $\alpha_i = \mathbb{P}[(s, a) \sim \pi^* | (s, a) \in D_i]$. This represents a demonstrator-level attribute: a higher α_i means demonstrator i tends to act more consistently w.r.t. π^*. We consider *demonstrator heterogeneity*, i.e., $\exists\, i, j \in [m], i \neq j$, such that $\alpha_i \neq \alpha_j$. The full dataset is $D = \{(i, D_i)\}_{i=1}^m$, and the expertise levels $\{\alpha_i\}_{i=1}^m$ are *unknown*.

[1] We provide further discussion and comparisons with related methods in Appendix A.

Optimality Score: For each (s, a) pair, we introduce a binary latent variable $z \in \{0, 1\}$ indicating its *optimality*: $z = 1 \Rightarrow (s, a) \sim \pi^*$, while $z = 0 \Rightarrow (s, a) \sim \pi^{\text{sub}}$, where π^{sub} denotes an unknown suboptimal policy. The *optimality score* is defined as the probability that (s, a) is optimal given its values: $\mathbb{P}(z = 1|s, a)$, which can be seen as a sample-specific, conditional expertise level.[2]

Let $\rho^{\pi^*}(s)$ and $\rho^{\pi^{\text{sub}}}(s)$ denote the state visitation distribution under π^* and π^{sub}, respectively. To capture demonstrator heterogeneity, we model demonstrations from each demonstrator-i as a statistical mixture[3]

$$\rho_i(s, a) = \alpha_i \rho^{\pi^*}(s)\, \pi^*(a \mid s) + (1-\alpha_i)\rho^{\pi^{\text{sub}}}(s)\pi^{\text{sub}}(a \mid s)$$

where $\rho^{\pi}(s, a) = \rho^{\pi}(s)\, \pi(a \mid s)$; $\rho^{\pi^*}(s, a)$ and $\rho^{\pi^{\text{sub}}}(s, a)$ are the occupancy measures of the optimal and suboptimal policies. This formulation corresponds to each demonstrator generating samples by first drawing $z \sim \text{Bernoulli}(\alpha_i)$ and then sampling $(s, a) \sim P(s, a \mid z)$. It ensures that all demonstrators share the same optimal and suboptimal occupancy components, differing only in expertise level α_i.

Policy Learning: Given demonstrations D from these heterogeneous demonstrators, our goal is to match the optimal policy π^*. Without access to rewards, we aim to learn a *scoring model*, parameterized by ϕ, $f_\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, where $\mathcal{S}$ denotes the set of states and $\mathcal{A}$ denotes the set of actions, to estimate the optimality scores $\mathbb{P}(z = 1|s, a)$ and then use it as a surrogate reward in RL: $\hat{\pi}^* = \text{argmax}_\pi \mathbb{E}_{p_\pi}\left[\sum_{t=1}^{T} \gamma^t f_\phi(s_t, a_t)\right]$.

3.2. Joint estimation of expertise and scoring model

We now describe how to learn the scoring model f_ϕ directly from the heterogeneous dataset D.

3.2.1. Surrogate demonstrator classification for optimality scoring

The optimality scoring task can be framed as a binary *class probability estimation* (Buja et al., 2005) problem, where we aim to estimate $\mathbb{P}(z = 1|s, a)$ for each (s, a) pair being optimal (with label $z = 1$) or sub-optimal (with label $z = 0$). In the supervised setting, if explicit optimality labels z were available, one could directly train f_ϕ with a proper scoring rule, such as cross-entropy. However, in our setting, such labels are not available. To address this challenge, we adapt the surrogate learning framework from the surrogate set classification (SSC) method of Lu et al. (2021).

Surrogate Demonstrator Classification We use the demonstrator identity $\bar{z} \in [m]$ as a surrogate label. Then the surrogate task involves training a multi-class classifier $\overline{f}_\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^m$ to predict the demonstrator from which each (s, a) pair originated, by minimizing the risk:

$$\mathcal{L}(\overline{f}_\phi) = \mathbb{E}_{(s,a)\sim\sum_i \mu_i \rho_i(s,a)}[\ell(\overline{f}_\phi(s, a), \bar{z})], \qquad (1)$$

where μ_i is the prior probability of sampling from demonstrator i, estimated as $\mu_i \approx \frac{||D_i||}{\sum_{i=1}^m ||D_i||}$, $||D_i||$ denotes the cardinality of the dataset D_i, and ℓ is the cross-entropy loss. For simplicity, we denote $\rho := \sum_i \mu_i \rho_i(s, a)$.

Recovering Optimality Scores Under demonstrator heterogeneity, Lu et al. (2021) show that the optimality scoring model f_ϕ can be recovered from the surrogate classifier $\overline{f}_\phi$ via an injective transformation $T(\cdot) : \mathbb{R} \to \mathbb{R}^m$, such that: $\overline{f}_\phi = T_i(f_\phi)$,[4] where the i-th component of T is

$$T_i(x) = \frac{\mu_i(\alpha_i - \alpha')x + \mu_i\alpha'(1 - \alpha_i)}{\sum_{i=1}^m \mu_i(\alpha_i - \alpha')x + \sum_{i=1}^m \mu_i\alpha'(1 - \alpha_i)}, \quad (2)$$

Here, $\alpha_i = \mathbb{P}(z = 1|\bar{z} = i)$ is the expertise level of demonstrator i, and α' is the expected expertise in the target environment where the scoring model is deployed. If these expertise levels are known, f_ϕ can be learned indirectly by training $\overline{f}_\phi$ via Eq. 1, which induces updates on f_ϕ through the following equivalent objective

$$\mathcal{L}(\phi) = \mathbb{E}_{(s,a)\sim\rho}\left[\ell(T(f_\phi(s, a)), \bar{z})\right]. \qquad (3)$$

Lemma 3.1 (Optimal Scoring Model Recovery, Lu et al. (2021))**.** *Assume the model class is well-specified, ℓ is a proper loss (e.g., cross-entropy), and T is correctly computed as in (2). Let $\overline{f}_\phi$ be the surrogate classifier trained optimally according to (3). Then the induced binary scoring model f_ϕ accurately estimates the optimality scores: $f_\phi(s, a) = \mathbb{P}(z = 1 \mid s, a)$.*

With the optimal scoring model, we can guarantee that the RL policy $\hat{\pi}^*$ learned using the surrogate reward f_ϕ matches the optimal policy π^* in both discounted occupancy and action distribution.

Theorem 3.2 (Optimal Policy Recovery)**.** *Let π^* be the unknown optimal policy. Define the discounted occupancy of any policy π by $\rho^\pi(s, a) =$*

[2] Our expertise modeling is twofold: α_i represents a demonstrator's overall skill (general expertise), while the optimality score captures state-action specific expertise (in-context expertise).

[3] We note that $\rho_i(s, a)$ is a *statistical* mixture over demonstrator-generated samples, analogous to the mixing proportions in Lu et al. (2019; 2020), rather than the occupancy induced by executing π_i in the MDP. In particular, its marginal $\rho_i(s) = \sum_a \rho_i(s, a) = \alpha_i\rho^{\pi^*}(s) + (1 - \alpha_i)\rho^{\pi^{\text{sub}}}(s)$ generally differs across demonstrators and is *not* equal to the MDP visitation distribution $\rho^{\pi_i}(s)$. This mixture model is used only to describe demonstration generation process, not to define the learner's policy.

[4] With a slight abuse of notation, we use f_ϕ and $\overline{f}_\phi$ to refer to both classifiers and class probability estimators, i.e., scoring functions in the sense of Buja et al. (2005).

Algorithm 1 EM-style Joint Learning of Expertise Levels and Scoring Model (Offline)

Input: $D = \{(i, D_i)\}_{i=1}^m$, number of queries n
repeat
  Initialization: Scoring model $f_\phi(s,a)$, expertise levels $\{\alpha_i\}_{i=1}^m \sim \mathcal{U}(0.1,\, 0.9),\ \forall i \in [m]$
  for $t = 1, 2, ..., T$ **do**
    M-step (Minimize risk to update $f_\phi(s,a)$):
    Update f_ϕ by minimizing the empirical risk from Eq. 3 using current $\{\alpha_i\}_{i=1}^m$
    E-step (Estimate $\{\alpha_i\}_{i=1}^m$):
    Update expertise levels $\alpha_i \leftarrow \frac{\sum_{(s,a)\in D_i}[f_\phi(s,a)>0.5]}{||D_i||},\ \forall i \in [m]$
  end for
until Expertise Dispersion Criterion is satisfied
Query n samples and correct predictions to satisfy **Optimality Alignment Criterion**
Return $\{\alpha_i\}_{i=1}^m$, f_ϕ

$\mathbb{E}_{\tau\sim p_\pi}\left[\sum_{t=1}^T \gamma^t \mathbf{1}_{\{(s_t,a_t)=(s,a)\}}\right]$, *where* $\mathbf{1}_{\{\cdot\}}$ *is the indicator function. Let the surrogate RL objective be*

$$J_f(\pi) = \mathbb{E}_{p_\pi}\left[\sum_{t=1}^T \gamma^t f_\phi(s_t,a_t)\right] = \sum_{s,a} \rho^\pi(s,a) f_\phi(s,a).$$

Assume the scoring model is optimal, i.e., $f_\phi(s,a) = \mathbb{P}(z = 1 \mid s,a)$ *for all* (s,a)*. Then any maximizer* $\hat{\pi}^* = \text{argmax}_\pi J_f(\pi)$ *satisfies* $\rho^{\hat{\pi}^*}(s,a) = \rho^{\pi^*}(s,a)$ *for all* (s,a) *in the support of* ρ^{π^*}*. If, in addition, the demonstration distribution satisfies the support coverage condition* $\text{supp}(\rho^{\pi^*}) \subseteq \text{supp}(\rho)$*, then for every state* s *with* $\rho^{\pi^*}(s) > 0$*, the learned policy recovers the optimal policy:* $\hat{\pi}^*(a \mid s) = \pi^*(a \mid s)$ *for all* a *in the support of* $\rho^{\pi^*}(s,\cdot)$*.*

3.2.2. EM-STYLE OPTIMIZATION OF SCORING MODEL AND EXPERTISE ESTIMATES

However, neither optimality labels nor expertise are given in our setting. To jointly learn both, we propose an EM-style algorithm (Algorithm 1).[5]

The algorithm begins by randomly initializing the expertise levels $\{\alpha_i\}_{i=1}^m$ within the range $[0.1, 0.9]$. Given the initial $\{\alpha_i\}_{i=1}^m$, we update the scoring model f_ϕ using the surrogate objective Eq. 3. We treat the expected expertise level α' of the target environment as a hyperparameter and fix it to 0.5. This choice corresponds to evaluating the scoring model using the *balanced risk*, a criterion adopted in Tangkaratt et al. (2020) and shown to be robust to label noise (Charoenphakdee et al., 2019; Menon et al., 2015). We further justify this choice through ablation studies presented in Tab. 15-16, and Fig. 7 in the Appendix. Next, we refine the expertise levels by computing the fraction of samples from each demonstrator that are labeled as optimal by the current scoring model. This alternating update procedure is repeated for T epochs. The following theorem establishes convergence of the EM-style optimization over α and ϕ to a stationary point. The proof is given in Appendix D.

Theorem 3.3 (Convergence Guarantee)**.** *Consider the optimization of the objective* $\mathcal{L}(\phi,\alpha)$ *via iterative updates of the scoring model* f_ϕ *and the expertise levels* $\alpha = \{\alpha_i\}_{i=1}^m$*. Let* (ϕ^t, α^t) *be the parameters at iteration* t*. If updates follow:*

$$\phi^{t+1} = \arg\min_\phi \mathcal{L}(\phi^t, \alpha^{t+1}), \tag{4}$$

$$\alpha^{t+1} = \frac{\sum_{(s,a)\in D_i}[f_{\phi^t}(s,a) > 0.5]}{||D_i||} \tag{5}$$

where $\mathcal{L}(\phi,\alpha) = \mathbb{E}_{(s,a)\sim\rho}\left[\ell(T_\alpha(f_\phi(s,a)), \bar{z})\right]$*, then the sequence* $\{\mathcal{L}(\phi^t,\alpha^t)\}_{t=1}^\infty$ *is monotonically non-increasing, i.e.,* $\mathcal{L}(\phi^{t+1},\alpha^{t+1}) \le \mathcal{L}(\phi^t,\alpha^t)$*, ensuring convergence to a stationary point.*

While Theorem 3.3 guarantees convergence, the quality of the solution is sensitive to the initialization of $\{\alpha_i\}_{i=1}^m$. As with other EM-style procedures, it may converge to suboptimal solutions depending on the initialization. To mitigate this, we run the algorithm with multiple random initializations until the Expertise Dispersion Criterion is satisfied. We report the number of initializations required in Tab. 17 in the Appendix.

Expertise Dispersion Criterion Since our algorithm trains on self-labeled data, it is susceptible to error propagation and confirmation bias. In practice, some random initializations lead to degenerate solutions, where $f_\phi(s,a)$ becomes overly confident and the estimated expertise levels collapse to nearly identical values. To assess the reliability of the learned expertise levels $\{\alpha_i\}_{i=1}^m$, we measure their *dispersion* by computing the variance $\text{Var}(\{\alpha_i\}_{i=1}^m)$. If the variance falls below a small threshold $\epsilon > 0$, we consider the model to have failed in distinguishing between different expertise levels, yielding low-information, overfit predictions. Conversely, a higher variance suggests more informative and diverse estimates, suggesting the model has captured demonstrator heterogeneity. We therefore use this variance as a selection criterion to retain only solutions with sufficient dispersion in $\{\alpha_i\}_{i=1}^m$.

Optimality Alignment Criterion Due to the unsupervised nature of our setting, the learned scoring model may inadvertently invert the notion of optimality. In such cases, the estimated expertise levels α_i may effectively correspond to $1 - \alpha_i$. To detect and correct this issue, we query a small

[5] To make the expertise level estimation problem solvable, we assume the existence of anchor points in the demonstration dataset, i.e., data points that belong to class $z = 1$ and $z = 0$ almost surely.

Algorithm 2 Joint Learning of Scoring Model and Agent Policy with Demonstration Relabeling (Online)

1: **Input**: Dataset $D = \{(i, D_i)\}_{i=1}^m$, learned scoring model $f_\phi(s,a)$, estimated expertise levels $\{\hat{\alpha}_i\}_{i=1}^m$, ratio of pseudo-optimal samples k
2: **Initialization**: early_stop $\leftarrow$ False, agent policy π_θ, $D_{\text{opt}} \leftarrow \{(s,a) \mid f_\phi(s,a) > 0.5\}$
3: **for** $t = 1, 2, ..., T$ **do**
4: Sample trajectories from agent: $D_{\pi_\theta} \leftarrow \{(s,a) \mid (s,a) \sim \pi_\theta\}$
5: Update f_ϕ by minimizing the empirical risk (Eq. 6)
6: **if** not early_stop **then**
7: Relabel pseudo-optimal set: $D_{\text{opt}} \leftarrow$ top-$k(s,a) \in D$ by $f_\phi(s,a)$
8: Check **early stopping criterion** and update early_stop if necessary
9: **end if**
10: Update π_θ using RL with reward $\log(f_\phi(s,a))$
11: **end for**
12: **Return** final policy π_θ

subset of state–action pairs (in our experiments, as few as 5 samples suffice) and ask an expert to verify the correctness of their predicted labels. If the majority of results indicate a systematic inversion in predicted labels, we simply flip the predicted labels and correct the outputs as follows: $\alpha_i \leftarrow 1 - \alpha_i, f_\phi(s,a) \leftarrow 1 - f_\phi(s,a), \forall i \in [m],\ \forall (s,a) \in D$.

3.3. Imitation learning via scoring model

After learning the scoring model f_ϕ and estimating the expertise levels $\{\alpha\}_{i=1}^m$, we perform IL.

3.3.1. Progressive scoring model refinement via fine-grained sample selection

The scoring model f_ϕ is initially trained only on demonstration data, limiting its generalization to learner trajectories due to covariate shift (Chang et al., 2021). To improve its robustness, we refine f_ϕ during agent learning using three complementary terms: *demonstration discriminative loss* (Eq. 3), reflecting the original training signal; *agent-matching penalty*, discouraging the misclassification of learner-generated behaviors as optimal; and a *mixup regularizer* (Zhang et al., 2017) which interpolates between agent and demonstrator behaviors to smooth the scoring boundary and stabilize training. The resulting objective is:

$$\min_{f_\phi} \widehat{\mathbb{E}}_{(s,a)\sim\rho}\big[\mathcal{L}(f_\phi(s,a))\big] + \widehat{\mathbb{E}}_{(s',a')\sim\rho^{\pi_\theta}}\big[\log(f_\phi(s',a'))\big] + \widehat{\mathbb{E}}_{\substack{(s,a)\sim\hat{\rho}^{\pi^*} \\ (s',a')\sim\rho^{\pi_\theta}}}\big[\mathcal{L}_{\text{mixup}}(s,a,s',a')\big] \tag{6}$$

where $\widehat{\mathbb{E}}$ denotes empirical expectation; $\hat{\rho}^{\pi^*}$ represents pseudo-optimal data inferred from $D_{\text{opt}} \subseteq D$; and

$$\mathcal{L}_{\text{mixup}} = \lambda \log\big(1 - f_\phi(\tilde{s},\tilde{a})\big) + (1-\lambda)\log\big(f_\phi(\tilde{s},\tilde{a})\big),$$
$$\text{where } \tilde{s},\tilde{a} = \lambda(s,a) + (1-\lambda)(s',a'), \lambda \sim \mathcal{U}(0,1)$$

Early in training, the novice agent tends to produce suboptimal behaviors. The agent-matching penalty discourages f_ϕ from assigning high scores to these early samples, thereby enhancing its ability to distinguish between agent and demonstrator behavior and reducing covariate shift. In parallel, mixup acts as both data augmentation and regularization based on agent and expert data, yielding more stable and informative gradients, ultimately improving the effectiveness of policy learning in IL (Orsini et al., 2021). The full training procedure is given in Algorithm 2.

To further refine f_ϕ, we adopt a bootstrapping approach where f_ϕ is used to pseudo-label demonstration data D. At each iteration: (1) we extract a pseudo-optimal subset $D_{\text{opt}} \subseteq D$ by applying a *top-k selection method*, where we choose the lowest $\lfloor k \cdot ||D|| \rfloor$ ($\lfloor \cdot \rfloor$ denotes the floor function) state–action pairs according to their f_ϕ scores; (2) we generate new agent trajectories D_{π_θ} to represent ρ^{π_θ} ; (3) we update f_ϕ using the loss in Eq. 6 (Algorithm 2 Line 5), computed over D, D_{π_θ}, and D_{opt}; (4) we relabel D_{opt} using the updated f_ϕ (Algorithm 2 Line 7). This mutual reinforcement between scoring model refinement and pseudo-label improvement allows both components to jointly bootstrap toward higher fidelity.

3.3.2. Policy learning with scoring model as a surrogate reward

As training progresses, the agent begins to generate increasingly competent behaviors, which are nonetheless treated as suboptimal during scoring model refinement. At this stage, the scoring model f_ϕ functions similarly to a discriminator in the GAIL framework (Ho & Ermon, 2016), struggling to distinguish between optimal and agent-generated data. Despite this, our approach is fundamentally different from GAIL, which relies exclusively on optimal demonstrations: our method operates in an (almost) unsupervised setting, allowing f_ϕ to learn from offline suboptimal data from heterogeneous demonstrators. This makes our approach especially effective when demonstration quality is low.

As the agent improves, f_ϕ becomes increasingly confused, and its predictions become unreliable. To prevent the accumulation of incorrect labels that could mislead policy learning, we introduce an **early stopping criterion** for the relabeling process: let Δ_t denote the number of sample changes in D_{opt} between iterations t and $t-1$, counting both removed and newly added samples; if $\Delta_t < \delta$ for p consecutive steps, we stop relabeling and freeze D_{opt}. The updated f_ϕ is employed as a surrogate reward function for

training the agent policy π_θ (Line 2). Specifically, we use $\log f_\phi(s, a)$ as the reward signal in reinforcement learning. In practice, we update the policy using soft actor-critic (SAC; Haarnoja et al., 2018), alternating with scoring model refinement and pseudo-label updates.

4. Experiments

We test whether our algorithm can (1) train a scoring model on multiple sets of imperfect demonstration data collected from heterogeneous demonstrators in order to provide optimality labels and estimate demonstrator expertise levels, and (2) learn an optimal agent policy based on the scoring model and imperfect demonstrations.

To evaluate the robustness of our algorithm under widely varying and low demonstrator expertise, we design two experiments: (a) **General expertise test**, using demonstrations collected from varying expertise levels in a wide range $[0.1 - 0.9]$ (average optimal ratio 0.5), and (b) **Low expertise test**, with extremely low expertise levels in $[0.05 - 0.15]$ (average optimal ratio 0.1). Prior works mostly focused on cases where most demonstrations are optimal, with limited exploration of low-optimal-ratio settings, where their performance degrades significantly (Eysenbach et al., 2021; Wang et al., 2021; 2023). To address this gap, we specifically evaluate the low expertise setting. In both tests, datasets are constructed from six demonstrators (each providing 5,000 samples) with uniformly distributed expertise levels (see Tab. 6 in the Appendix). We set the top-k hyperparameter k to match the average optimal ratio in each test. Code for our experiments is available at `https://github.com/ch-engpan/IL_imperfect`.

4.1. Environment and datasets

We evaluate our algorithm against baseline methods on four MuJoCo environments (Todorov et al., 2012; Towers et al., 2024) and conduct ablation studies to assess the contributions of our proposed components. Demonstrations are collected by training a policy with SAC with the true reward. An early-stage checkpoint π^{sub} provides suboptimal demonstrations, while the final checkpoint π^* provides optimal demonstrations (see Tab. 5 in the Appendix for the checkpoints' performance). As in Beliaev et al. (2022); Tangkaratt et al. (2020), the imperfect dataset is mixture of state-action pairs from π^* and π^{sub}. Unlike random behaviors, our suboptimal data comes from a real suboptimal policy, which reflects that experts may make mistakes, but their actions are still better than random actions. Using this approach, multiple imperfect demonstration sets $D = \{(i, D_i)\}_{i=1}^m$ are collected.

4.2. Comparison with Baselines

Across the experiments, we benchmark our model against four IL baseline algorithms. (1) **GAIL o.s. (optimal subset)** (Ho & Ermon, 2016): GAIL trained on the purely optimal subset of demonstrations, serving as an oracle method; (2) **GAIL**: GAIL trained on all (optimal and suboptimal) demonstrations; (3) **RIL** (Tangkaratt et al., 2020): trains an optimality discriminator via pseudo-labeling with co-training; (4) **WGAIL** (Wang et al., 2021): estimates importance weights to emphasize near-optimal demonstrations; (5) **ILEED** (Beliaev et al., 2022): learns the policy in a offline training manner via maximum-likelihood; (6) **BC** (Pomerleau, 1988). For fair comparison, we adopt a consistent model architecture for discriminator/classifier/scoring model, and learner policy, modifying only the input and output layers to match the state and action spaces of each environment (Arulkumaran & Lillrank, 2024). Moreover, all online baselines incorporate mixup regularization. All learner policies are updated using SAC with same hyperparameters. Training details are provided in Tab. 19 in the Appendix. We use 5 random seeds per condition and report the mean and standard deviation in all our results. For offline methods (ILEED and BC), for each random seed, we repeat the algorithm 10 times with random initializations and report the best result by selecting the run with the maximal likelihood.

The results are summarized in Tab. 2 (see Fig. 3 for training curves, and Tab. 18 for more detailed results of offline methods in the Appendix). In the general expertise test, our algorithm outperforms all baselines except GAIL o.s., achieving at least twice the rewards in all tasks. While GAIL o.s. attains comparable rewards, our algorithm still yields slightly higher rewards in Ant, Swimmer and Walker2d. This is attributed to the fact that our algorithm can achieve high classification performance on imperfect demonstrations (see Tab. 4), and, in contrast to GAIL o.s., is also capable of leveraging suboptimal data to perform better in the subsequent online IL stage. In the low expertise test, the trends remain similar: our algorithm surpasses all baselines except GAIL o.s. However, decreased classification performance on imperfect demonstrations (see Tab. 4) reduces overall rewards compared to the general expertise test. In this setting, GAIL o.s., which trains on purely optimal subset of demonstrations, achieves higher rewards in most tasks.

The low rewards achieved by other baselines in both experiments can be attributed to the several factors. For GAIL, which treats all demonstrations as optimal, its performance is limited when handling imperfect demonstrations. RIL assumes that most demonstrations are optimal, but in our experiments, at least half are suboptimal. Consequently, the discriminator's classification risk does not align with

Table 2. Performance comparison with baselines, mean $\pm$ std of reward across last five checkpoints

Environment	Dataset	GAIL o.s.	BC	ILEED	GAIL	RIL	WGAIL	LEOS-IL (ours)
General expertise test								
Ant-v4	4016.3 ± 2776.8	5660.4 ± 163.6	1341.8 ± 164.5	1428.2 ± 80.3	1387.2 ± 301.0	101.8 ± 118.0	1744.0 ± 919.1	$\mathbf{6084.6 \pm 146.9}$
HalfCheetah-v4	9988.9 ± 5959.9	9278.2 ± 496.5	2318.1 ± 521.2	1526.2 ± 1020.4	3035.8 ± 1186.4	1631.6 ± 959.7	4336.2 ± 335.6	$\mathbf{8751.2 \pm 1797.9}$
Swimmer-v4	190.7 ± 149.1	293.6 ± 28.2	21.3 ± 23.0	16.8 ± 20.5	76.6 ± 28.3	28.8 ± 16.3	115.6 ± 83.7	$\mathbf{305.4 \pm 15.7}$
Walker2d-v4	3096.7 ± 2549.6	3703.0 ± 1011.9	308.4 ± 111.5	440.3 ± 313.0	948.6 ± 297.2	1187.2 ± 130.1	665.4 ± 330.2	$\mathbf{3959.2 \pm 671.4}$
Low expertise test								
Ant-v4	1816.4 ± 1726.3	5542.0 ± 168.1	1473.4 ± 97.7	1376.5 ± 62.1	1439.6 ± 202.0	2.2 ± 174.1	971.6 ± 591.0	$\mathbf{5932.0 \pm 335.2}$
HalfCheetah-v4	5222.0 ± 3578.8	9194.2 ± 848.1	3753.9 ± 220.5	3767.6 ± 176.9	3162.8 ± 1417.2	964.2 ± 1100.1	3975.6 ± 369.9	$\mathbf{7335.0 \pm 1316.4}$
Swimmer-v4	71.4 ± 89.5	244.4 ± 113.5	27.8 ± 11.6	34.0 ± 2.4	37.4 ± 10.5	44.6 ± 2.9	48.8 ± 53.7	$\mathbf{195.4 \pm 92.1}$
Walker2d-v4	1062.8 ± 1546.5	3532.6 ± 782.2	436.6 ± 69.1	605.3 ± 94.5	737.8 ± 163.2	1140.0 ± 319.0	703.8 ± 291.2	$\mathbf{2961.8 \pm 339.8}$

Table 3. Estimation errors of Stage 1 scoring model for expertise levels (mean $\pm$ std).

Experiment	Ant-v4	HalfCheetah-v4	Hopper-v4	Swimmer-v4	Walker2d-v4
General expertise test	$(2.67 \pm 1.28) \times 10^{-4}$	$(2.00 \pm 2.00) \times 10^{-5}$	$(3.83 \pm 2.73) \times 10^{-3}$	$(2.00 \pm 2.00) \times 10^{-5}$	$(3.27 \pm 3.12) \times 10^{-4}$
Low expertise test	$(9.65 \pm 0.24) \times 10^{-2}$	$(3.67 \pm 0.15) \times 10^{-2}$	$(1.05 \pm 0.02) \times 10^{-1}$	$(2.81 \pm 0.11) \times 10^{-2}$	$(1.34 \pm 0.04) \times 10^{-1}$

the risk of distinguishing between optimal and suboptimal demonstrations, reducing the reliability of its predictions for policy learning. Moreover, even with co-training for pseudo-labeling, the overconfidence can only be partially mitigated; once error accumulation begins, it continues to degrade the discriminator's ability to predict optimality of demonstrations. WGAIL requires an optimal discriminator to compute importance weights, assuming the early-stage GAIL discriminator is near-optimal. Although early stopping is used for weight estimation, this assumption can fail due to the instability of adversarial learning. ILEED is sensitive to state embedding quality and assumes uniform state coverage, making it prone to biased expertise estimates under sparse data. Being offline and MLE-based, it is vulnerable to distribution shift, and its tightly coupled joint optimization can lead to instability.

4.3. Estimation of expertise levels

Tab. 3 reports the estimation errors of the scoring model trained in Stage 1 w.r.t. expertise levels relative to the ground truth, with detailed results provided in Tab. 7-8 in the Appendix. In the general expertise test, where true expertise levels are widely distributed within [0.1-0.9], errors are extremely low, remaining below 0.004 across all tasks. In contrast, in the low expertise test, with levels [0.05-0.15], the estimation errors increase to around 0.1 for most tasks. This indicates that the scoring model tends to be over-optimistic and produce more false positive (FP) labels on imperfect demonstrations (see Tab. 8). However, these incorrect labels are largely corrected during Stage 2.

4.4. Optimality Prediction on Imperfection Demonstrations

We evaluate the scoring model f_ϕ on its ability to classify imperfect demonstrations by optimality. Labeling is performed in two stages: Stage 1 uses pretrained f_ϕ and Stage 2 uses fine-tuned f_ϕ. Performance is measured using accuracy and precision, with a focus on precision, as it reflects the FP risk, which is more critical than false negative risk for policy learning (Irpan et al., 2019).

Results in Tab. 4 show that in the general expertise test, both accuracy and precision are nearly 1 in Stage 1, leaving almost no room for improvement in Stage 2. In contrast, in the low expertise test, where expertise levels are narrowly distributed, the accuracy in Stage 1 stays above 86%, but the precision drops significantly, to 43-78%. This indicates that surrogate demonstrator classification training handles datasets with widely varying expertise well, but struggles when expertise levels are similar. However, Stage 2 improves both accuracy and precision, with precision increasing by 12-115% across tasks. This shows that for imperfect demonstrations with extremely low expertise, many false positives from a poorly pretrained f_ϕ can be corrected in Stage 2 after fine-tuning f_ϕ. Detailed classification results are presented in Tab. 8 in the Appendix.

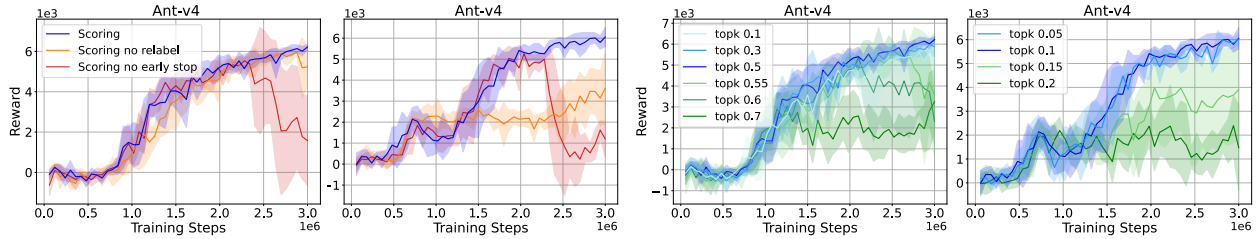
4.5. Ablation tests

To assess the effect of each component of our algorithm, we conduct ablation experiments on the Ant-v4 environment.

Relabeling and Early Stopping: We evaluate the effects of the relabeling step and early stopping in Stage 2 (Alg. 2), shown in Fig. 2 (a)-(b) (see Tab. 9 in the Appendix for numerical results). Without relabeling, performance is comparable to the original method in the general expertise test but drops by about half in the low expertise test due to FP errors generated in Stage 1 (see Sec. 4.4), which cannot be corrected without Stage 2 relabeling. Removing early stopping degrades rewards in both general and low expertise tests as the agent approaches the optimal policy. Near-optimal agent samples confuse f_ϕ, increasing classification errors and accumulating incorrect labels, which reinforces overconfidence of f_ϕ and corrupts the surrogate reward for

Table 4. Classification results of the scoring model on imperfect demonstrations for Stage 1 and Stage 2 (Accuracy / Precision, %), with Stage 1 → Stage 2 improvement (%) after relabeling.

Test	Stage	Ant-v4	Impr.	HalfCheetah-v4	Impr.	Swimmer-v4	Impr.	Walker2d-v4	Impr.
General expertise	1	99.91 / 99.89	-	100.00 / 100.00	-	100.00 / 100.00	-	99.92 / 99.90	-
	2	99.89 / 99.89	-0.02 / 0.00	100.00 / 100.00	0.00 / 0.00	99.94 / 99.94	-0.06 / -0.06	99.94 / 99.94	0.02 / 0.04
Low expertise	1	90.33 / **51.78**	-	96.33 / **73.68**	-	97.19 / **78.84**	-	86.39 / **43.01**	-
	2	99.65 / **98.23**	10.32 / **89.71**	99.98 / **99.91**	3.79 / **35.60**	97.58 / **87.91**	0.40 / **11.50**	98.48 / **92.40**	13.99 / **114.83**



(a) General expertise test (b) Low expertise test (c) General expertise test (d) Low expertise test

Figure 2. Ablation results on the impact of Stage 2 relabeling and early stopping, and k selection

the agent.

Top-k k Value Selection: We study the effect of different k values on IL performance . Results are shown in Fig. 2(c–d), Tab. 10. For both the general and low expertise tests, (average optimality ratios $k^* = 0.5$ and 0.1), rewards remain stable when $k \leq k^*$ but decline for $k > k^*$. The reason is that the pseudo optimal set is selected based on the lowest $\lfloor k \cdot ||D|| \rfloor$ state–action pairs by f_ϕ scores. Assuming f_ϕ is well trained, a $k < k^*$ leads to a conservative selection (subset of the true optimal set), while $k > k^*$ can introduce more FP samples to the pseudo optimal set. This indicates that FP risk is more critical, and a smaller k is preferable.

Number of Demonstrators: We ablate the number of demonstrators and find that our method is robust to varying counts in terms of IL performance (see Fig. 4 and Tab. 11-12 in the Appendix). Appendix E.2 shows that our method consistently matches or outperforms variants (PN loss for discrimination, oracle settings with given expertise levels or optimality labels), highlighting its robustness and effectiveness.

Number of human queries: We analyze sensitivity to the number of human queries. Results (Tab. 13-14 and Fig. 6 in the Appendix) show that even a single query performs comparably to five queries, due to the very high Stage 1 classification accuracy (>99%) and similar reward performance across query settings. Nevertheless, queries remain necessary to assign correct labels to clusters in the absence of ground-truth annotations.

Expected Expertise α': We analyze the effect of different values of α' (Tab. 15-16 and Fig. 7 in the Appendix). Performance peaks near $\alpha' = 0.5$ and drops when α' is very small or large, indicating that $\alpha' = 0.5$ is optimal for both the general and low expertise tests.

5. Conclusions

We presented a general IL framework for handling unlabeled demonstrations from heterogeneous imperfect demonstrators with unknown expertise. Our approach combines EM-style training with a refinement step to learn a scoring model that guides policy learning. Experiments across challenging settings demonstrate strong performance, especially in low-expertise regimes, highlighting the value of our method's ability to leverage suboptimal data in an (almost) unsupervised manner.

Impact Statement

This paper presents work whose goal is to advance the field of machine learning. There are many potential societal consequences of our work, particularly when applied to large-scale training on extensive datasets, such as language models or vision-language-action models for robots. Large-scale datasets can contain suboptimal data, and our approach can improve performance compared to treating all data as equally optimal for supervision. We do not identify any specific societal risks that require highlighting at this time.

Acknowledgements

This work was partially supported by JST under Moonshot R&D Grant Number JPMJMS2012.

References

Arulkumaran, K. and Lillrank, D. O. A pragmatic look at deep imitation learning. In *Asian Conference on Machine Learning*, pp. 58–73. PMLR, 2024.

Beliaev, M., Shih, A., Ermon, S., Sadigh, D., and Pedarsani,

R. Imitation learning by estimating expertise of demonstrators. In *International Conference on Machine Learning*, pp. 1732–1748. PMLR, 2022.

Belkhale, S., Cui, Y., and Sadigh, D. Data quality in imitation learning. *Advances in neural information processing systems*, 36:80375–80395, 2023.

Brown, D., Goo, W., Nagarajan, P., and Niekum, S. Extrapolating beyond suboptimal demonstrations via inverse reinforcement learning from observations. In *International conference on machine learning*, pp. 783–792. PMLR, 2019.

Brown, D. S., Goo, W., and Niekum, S. Better-than-demonstrator imitation learning via automatically-ranked demonstrations. In *Conference on robot learning*, pp. 330–359. PMLR, 2020.

Buja, A., Stuetzle, W., and Shen, Y. Loss functions for binary class probability estimation and classification: Structure and applications. *Technical report*, 2005.

Cai, X.-Q., Zhang, Y.-J., Chiang, C.-K., and Sugiyama, M. Imitation learning from vague feedback. *Advances in Neural Information Processing Systems*, 36:48275–48292, 2023.

Chang, J., Uehara, M., Sreenivas, D., Kidambi, R., and Sun, W. Mitigating covariate shift in imitation learning via offline data with partial coverage. *Advances in Neural Information Processing Systems*, 34:965–979, 2021.

Charoenphakdee, N., Lee, J., and Sugiyama, M. On symmetric losses for learning from corrupted labels. In *International Conference on Machine Learning*, pp. 961–970. PMLR, 2019.

Eysenbach, B., Levine, S., and Salakhutdinov, R. Replacing rewards with examples: Example-based policy search via recursive classification. *arXiv preprint arXiv:2103.12656*, 2021.

Fu, J., Luo, K., and Levine, S. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Ho, J. and Ermon, S. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

Irpan, A., Rao, K., Bousmalis, K., Harris, C., Ibarz, J., and Levine, S. Off-policy evaluation via off-policy classification. *Advances in Neural Information Processing Systems*, 32, 2019.

Lee, G., Oh, W., Oh, J., Shin, S., Kim, D., Jeong, J., Choi, S., and Oh, S. Semi-supervised imitation learning with mixed qualities of demonstrations for autonomous driving. In *2022 22nd International Conference on Control, Automation and Systems (ICCAS)*, pp. 20–25. IEEE, 2022.

Lu, N., Niu, G., Menon, A. K., and Sugiyama, M. On the minimal supervision for training any binary classifier from only unlabeled data. In *ICLR*, 2019.

Lu, N., Zhang, T., Niu, G., and Sugiyama, M. Mitigating overfitting in supervised classification from two unlabeled datasets: A consistent risk correction approach. In *AISTATS*, 2020.

Lu, N., Lei, S., Niu, G., Sato, I., and Sugiyama, M. Binary classification from multiple unlabeled datasets via surrogate set classification. In *International Conference on Machine Learning*, pp. 7134–7144. PMLR, 2021.

Menon, A., Van Rooyen, B., Ong, C. S., and Williamson, B. Learning from corrupted binary labels via class-probability estimation. In *International conference on machine learning*, pp. 125–134. PMLR, 2015.

Orsini, M., Raichuk, A., Hussenot, L., Vincent, D., Dadashi, R., Girgin, S., Geist, M., Bachem, O., Pietquin, O., and Andrychowicz, M. What matters for adversarial imitation learning? *Advances in Neural Information Processing Systems*, 34:14656–14668, 2021.

Pomerleau, D. A. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

Ramaswamy, H., Scott, C., and Tewari, A. Mixture proportion estimation via kernel embeddings of distributions. In *International conference on machine learning*, pp. 2052–2060. PMLR, 2016.

Saxena, V., Bronars, M., Arachchige, N. R., Wang, K., Shin, W. C., Nasiriany, S., Mandlekar, A., and Xu, D. What matters in learning from large-scale datasets for robot manipulation. *arXiv preprint arXiv:2506.13536*, 2025.

Shangnan, Z. and Wang, Y. Quantum cross entropy and maximum likelihood principle. *arXiv preprint arXiv:2102.11887*, 2021.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Tangkaratt, V., Charoenphakdee, N., and Sugiyama, M. Robust imitation learning from noisy demonstrations. *arXiv preprint arXiv:2010.10181*, 2020.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Towers, M., Kwiatkowski, A., Terry, J., Balis, J. U., De Cola, G., Deleu, T., Goulão, M., Kallinteris, A., Krimmel, M., KG, A., et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

Wang, Y., Xu, C., Du, B., and Lee, H. Learning to weight imperfect demonstrations. In *International Conference on Machine Learning*, pp. 10961–10970. PMLR, 2021.

Wang, Y., Du, B., and Xu, C. Unlabeled imperfect demonstrations in adversarial imitation learning. *arXiv preprint arXiv:2302.06271*, 2023.

Wu, Y.-H., Charoenphakdee, N., Bao, H., Tangkaratt, V., and Sugiyama, M. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pp. 6818–6827. PMLR, 2019.

Zare, M., Kebria, P. M., Khosravi, A., and Nahavandi, S. A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, 2024.

Zhang, C., Wang, W., Chen, Z., and Xi, J. 100 drivers, 2200 km: A natural dataset of driving style toward human-centered intelligent driving systems. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pp. 351–356. IEEE, 2024.

Zhang, H., Cisse, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Appendix

A. Further Discussion on Related Work

Closer comparisons with ILEED (Beliaev et al., 2022) ILEED is the closest work to ours, as both address IL from multiple demonstrators with heterogeneous expertise. Despite the shared high-level motivation, the two approaches differ substantially in problem setting, modeling assumptions, methodologies and robustness properties, making them suitable for different practical regimes.

- **Problem setting.** ILEED considers the problem setting where each demonstrator's policy is a mixture of the optimal policy and a *uniformly random* policy, with the mixing coefficient interpreted as an expertise level.[6] While convenient, this "suboptimal = uniform noise" assumption is restrictive as real-world suboptimal behavior is typically structured, rather than near-uniform noise. For example, a demonstrator may consistently struggle with specific skills without acting randomly overall.

  In contrast, we relax this assumption by modeling each demonstrator's behavior as a mixture of the optimal policy and an *unknown* suboptimal policy. Our formulation imposes no uniformity or high-entropy constraints on the suboptimal component, enabling the method to capture realistic and highly structured forms of suboptimality.

- **Expertise modeling.** ILEED models expertise as a state-dependent function defined as a sigmoid of the inner product between the learned state and demonstrator embeddings. Although this formulation can represent state-varying expertise, it relies heavily on the quality of the recovered embeddings for *all* states visited by *all* demonstrators. In practice, when demonstrators explore only partial or disjoint regions of the state space, the resulting embeddings, and thus the inferred expertise levels, can become severely biased, causing significant policy degradation.

  Our method instead models expertise through the class-prior probability $\mathbb{P}(z{=}1)$, estimated via a surrogate scoring model that directly distinguishes optimal from suboptimal samples across demonstrators. This approach does not require each demonstrator to provide full coverage of the state space. Moreover, the trained scoring model outputs a sample-wise optimality score $\mathbb{P}(z{=}1 \mid s, a)$ that can be used as a reward to train an RL agent (SAC).

- **Methodology.** ILEED performs fully *offline* maximum-likelihood estimation of the policy, state embeddings, and demonstrator embeddings. As a result, it is sensitive to *distribution shift* between the demonstration distribution and the agent's on-policy distribution during training.

  Our method instead employs a *two-stage* procedure: we first learn a scoring model offline, then treat its log-outputs as a surrogate reward for an RL agent and *iteratively refine* the scorer using on-policy rollouts. To explicitly mitigate covariate shift, we incorporate mixup augmentation, an agent-matching penalty, and a relabeling mechanism, making our approach more robust when the agent visits states that rarely or never appear in the demonstrations.

- **Optimization stability.** ILEED jointly optimizes policy parameters, state embeddings, and demonstrator embeddings via a single maximum-likelihood objective. This fully coupled optimization can be unstable, as conflicting gradients among latent components may lead to degenerate solutions.

  Our method instead employs an EM-style alternating optimization for expertise estimation and scoring-model learning, for which we provide a convergence guarantee. Policy learning then proceeds through a standard RL pipeline using the scoring model as a reward proxy, yielding considerably more stable optimization behavior in practice.

Our empirical results (see Tab. 2, Fig. 3, Tab. 18) validate these distinctions. In particular, our method is advantageous when demonstrations contain structured suboptimality, exhibit low expertise levels, or when distribution shift between demonstrations and agent rollouts is significant.

Comparisons with T-REX/D-REX (Brown et al., 2019; 2020) Both *trajectory-ranked reward extrapolation* (T-REX) and *disturbance-based reward extrapolation* (D-REX) learn a reward function from ranked demonstrations, allowing an agent to potentially outperform the demonstrator via extrapolation. T-REX relies on experts to provide ranking comparisons,

[6] For continuous action spaces, ILEED replaces this uniform noise component with a high-entropy variant of the optimal policy. Specifically, the suboptimal component is modeled by scaling the variance of a Gaussian mixture policy, resulting in a near-uniform distribution rather than a structured suboptimal policy.

while D-REX can automatically generate rankings by injecting noise into a cloned policy, based on the assumption that added noise leads to a continuous degradation in performance.

Compared to T-REX, our method does not require large amounts of supervision or expert ranking feedback during training. Moreover, while D-REX assumes homogeneous demonstrations and introduces heterogeneity artificially, our approach naturally handles heterogeneous datasets and is capable of inferring varying expertise levels within the data. Unlike D-REX, our method does not rely on the assumption that policy performance monotonically decreases with noise injection, which may not hold in more complex domains. In addition, both T-REX and D-REX first learn a surrogate reward function and then optimize a policy using RL on this fixed reward, which can lead to covariate shift when the agent encounters states not present in the demonstrations. Our method addresses this issue by continuously fine-tuning the scoring model with new data collected during RL, resulting in improved robustness and generalization. Overall, our approach provides a flexible and practical framework for leveraging demonstrations, particularly in settings with heterogeneous data and limited supervision.

Comparisons with VPIL (Cai et al., 2023) *Vaguely pairwise imitation learning* (VPIL) investigates a weak-supervision setting in which the annotator provides vague pairwise feedback: a pair of demonstrations is labeled only when their quality differs substantially, resulting in two contaminated datasets that still contain mixtures of expert and non-expert trajectories. VPIL leverages this pairwise structure to recover the expert occupancy measure through *mixture-proportion estimation* (MPE; Ramaswamy et al., 2016) and risk rewriting, and its *comparative imitation learning with risk rewriting with estimation* (COMPILER-E) algorithms integrate the recovered distribution into a GAIL-style adversarial IL framework.

Our problem setting is more general and arguably more challenging. Unlike VPIL, which assumes access to vague pairwise comparison labels, we consider a scenario in which the agent receives only heterogeneous demonstrations of unknown and varying quality, with no preference, ranking, or optimality labels. In the absence of such supervision, the learner must infer both the underlying demonstration quality and the reward surrogate directly from unlabeled trajectories.

The methodological differences therefore extend beyond supervision assumptions. Rather than reconstructing an expert occupancy measure from pairwise datasets, we learn a fine-grained state–action optimality scoring model through a surrogate demonstrator classification objective and an EM-style latent expertise level estimation procedure. The trained scoring model then serves as a surrogate reward within an iterative policy-learning loop, enabling robust imitation learning directly from heterogeneous, unlabeled demonstrators without relying on any preference-based supervision.

Comparisons with AIRL (Fu et al., 2017) *Adversarial inverse reinforcement learning* (AIRL) extends the GAIL framework by structuring the discriminator to decompose into a state-only reward term and a shaping term, enabling recovery of a reward function that is disentangled from environment dynamics. This structure allows AIRL to learn portable rewards that can be re-optimized in new or modified environments.

Our problem setting, however, is fundamentally different. Rather than assuming demonstrations come from a single (near-optimal) expert, we address imitation learning from heterogeneous, unlabeled, and potentially highly suboptimal demonstrators with unknown expertise levels. Although our method learns a scoring model that serves as a surrogate reward, the objective is not reward recovery or transfer, but robust policy learning when expert dominance does not hold. AIRL, like GAIL, is not designed to handle heterogeneous demonstrators and presumes access to expert demonstrations. Therefore, their assumptions and objectives cannot address the key challenges in our setting.

B. Surrogate Demonstrator Classification training

We provide more details about the algorithm and theory of Surrogate Demonstrator Classification training.

We consider demonstrator heterogeneity, i.e., there $\exists\, i, j \in [m], i \neq j$, such that $\alpha_i \neq \alpha_j$. According to Lu et al. (2021), the $f_{\phi^*_{\text{surr}}}$ that is recovered by removing transformation layer $T(\cdot)$ of the optimal surrogate classifier $\overline{f}_{\phi^*}$ (the minimizer of $\mathbb{E}_{(s,a)\sim\rho}[\ell(\overline{f}_\phi(s,a),\bar{z})] = \mathbb{E}_{(s,a)\sim\rho}\left[\ell(T(f_\phi(s,a)),\bar{z})\right]$) is identical to the optimal classifier f_{ϕ^*} that minimizes the risk $\mathbb{E}_{(s,a)\sim\rho}\left[\ell(f_\phi(s,a),z)\right]$, which is not possible if the heterogeneity setting is invalid. Thus, through the transformation function transformation $T(\cdot)$, the optimality scoring model f_{ϕ^*} can be learned by minimizing the surrogate classification risk in Eq. 3.

Algorithm 3 Learning scoring model

Input: $\{(i, D_i)\}_{i=1}^m$, $\{\alpha_i\}_{i=1}^m$, and α'
Initialization: scoring model $f_\phi(s, a)$
Compute of $T(\cdot) = [T_1(\cdot), \cdots, T_m(\cdot)]$ using Eq. 2 based on $\{\alpha_i\}_{i=1}^m$ and α'
for $t = 1, 2, \cdots$ **do**
  Sample examples: $\bar{z}, (s, a) \sim \{(i, D_i)\}_{i=1}^m$
  Update f_ϕ by minimizing the empirical risk from Eq. 3
end for
Return f_ϕ

C. Proof of Theorem 3.2

Proof. Let $c = \mathbb{P}(z = 1)$ be the optimality prior probability in the dataset. By Bayes' rule, we have

$$f(s, a) = \mathbb{P}(z = 1 \mid s, a) = c\frac{\rho^{\pi^*}(s, a)}{\rho(s, a)}.$$

Thus, the surrogate RL objective can be written as

$$J_f(\pi) = \sum_{s,a} \rho^\pi(s, a) f_\phi(s, a) = c \sum_{s,a} \frac{\rho^\pi(s, a)\rho^{\pi^*}(s, a)}{\rho(s, a)}.$$

Define the following vectors indexed by (s, a): $x_{s,a} = \frac{\rho^\pi(s,a)}{\sqrt{\rho(s,a)}}$ and $y_{s,a} = \frac{\rho^{\pi^*}(s,a)}{\sqrt{\rho(s,a)}}$, then we have

$$\begin{aligned} J_f(\pi) &= c\langle x, y\rangle \\ &\leq c\|x\|_2\|y\|_2 \\ &= c\sqrt{\sum_{s,a} \frac{\rho^\pi(s, a)^2}{\rho(s, a)}}\sqrt{\sum_{s,a} \frac{\rho^{\pi^*}(s, a)^2}{\rho(s, a)}}, \end{aligned} \tag{7}$$

where we applied the Cauchy-Schwarz inequality in (7). The above equality holds if and only if $x = \beta y$ for some scalar $\beta \geq 0$, that is,

$$\frac{\rho^\pi(s, a)}{\sqrt{\rho(s, a)}} = \beta\frac{\rho^{\pi^*}(s, a)}{\sqrt{\rho(s, a)}},\ \forall (s, a) \text{ in } \mathrm{supp}(\rho^\pi).$$

Since $\sqrt{\rho(s, a)} > 0$, the above equality reduces to

$$\rho^\pi(s, a) = \beta\rho^{\pi^*}(s, a),\ \forall (s, a) \text{ in } \mathrm{supp}(\rho^\pi).$$

Summing both sides over (s, a), and given the fact that both ρ^π and ρ^{π^*} have the same total mass $\sum_{s,a} \rho^\pi(s, a) = \sum_{t=1}^T \gamma^t > 0$, we obtain that $\beta = 1$, and therefore $\rho^\pi = \rho^{\pi^*}$. This proves the optimal occupancy recovery.

If $\mathrm{supp}(\rho^{\pi^*}) \subseteq \mathrm{supp}(\rho)$, then for any state s with $\rho^{\pi^*}(s) > 0$, we have $\rho^{\hat{\pi}^*}(s) > 0$. The conditional action probabilities are then recovered by normalization

$$\hat{\pi}^*(a \mid s) = \frac{\rho^{\hat{\pi}^*}(s, a)}{\rho^{\hat{\pi}^*}(s)} = \frac{\rho^{\pi^*}(s, a)}{\rho^{\pi^*}(s)} = \pi^*(a \mid s),$$

where the second equality is due to $\rho^{\hat{\pi}^*} = \rho^{\pi^*}$. Thus $\hat{\pi}^*$ matches π^* for every state s in the support of ρ^{π^*}. □

D. Proof of Theorem 3.3

Proof. We first prove that the sequence $\{\mathcal{L}(\phi^t, \alpha^t)\}$ is non-increasing. If $\mathcal{L}(\phi, \alpha)$ is lower-bounded (e.g., by zero in many loss formulations), then by the monotone convergence theorem, the sequence must converge.

$$\mathcal{L}(\phi, \alpha) = \mathbb{E}_{(s,a)\sim\rho}\left[\ell(T_\alpha(f_\phi(s,a)), \bar{z})\right] \tag{8}$$

Minimizing the cross-entropy loss in Eq. 8 is equivalent to maximizing a log likelihood (Shangnan & Wang, 2021). We define a log likelihood function through the conditional distribution $p(\bar{z}|(s,a))$ in the form of discriminative training. With scoring model parameters and hidden parameters of expertise levels, the log likelihood (ll) can be formulated as

$$ll(\phi, \alpha) := \ln p(\bar{z}|(s,a), \phi) = \sum_\alpha p(\alpha)\ \ln p(\bar{z}|(s,a), \phi) \tag{9}$$

To prove that $\{\mathcal{L}(\phi^t, \alpha^t)\}$ is monotonically non-increasing, we only have to prove $\{ll(\phi^t, \alpha^t)\}$ is monotonically non-decreasing, i.e., $ll(\phi^{t+1}, \alpha^{t+1}) \geq ll(\phi^t, \alpha^t)$.

We define $L(q, \phi) := \sum_\alpha q(\alpha) \ln \frac{p(\bar{z}, \alpha|(s,a), \phi)}{q(\alpha)}$, $q := q(\alpha)$, and $p := p(\alpha|\bar{z}, (s,a), \phi)$

$$\begin{aligned}
L(q, \phi) &= \sum_\alpha q(\alpha) \ln \frac{p(\bar{z}, \alpha|(s,a), \phi)}{q(\alpha)} \\
&= \sum_\alpha q(\alpha) \left[\ln p(\alpha|\bar{z}, (s,a), \phi) + \ln p(\bar{z}|(s,a), \phi) - \ln q(\alpha)\right] \\
&= \sum_\alpha q(\alpha) \ln \frac{p(\alpha|\bar{z}, (s,a), \phi)}{q(\alpha)} + \sum_\alpha q(\alpha) \ln p(\bar{z}|(s,a), \phi) \\
&= -KL(q||p) + \ln p(\bar{z}|(s,a), \phi)
\end{aligned} \tag{10}$$

Then, the likelihood $ll(\phi, \alpha)$ can be represented as

$$\begin{aligned}
\ln p(d|(s,a), \phi) &= L(q, \phi) + KL(q||p), \\
&\geq L(q, \phi)
\end{aligned} \tag{11}$$

Since $KL(q||p) \geq 0$, $L(q, \phi)$ is the lower bound of $ll(\phi, \alpha)$.

For likelihood function $ll(\phi^t, \alpha^t)$ at iteration t,

$$\begin{aligned}
ll(\phi^t, \alpha^t) &= \ln p(\bar{z}|(s,a), \phi^t) \\
&= L(q(\alpha^t), \phi^t) + KL(q(\alpha^t)||p(\alpha|\bar{z}, (s,a), \phi^t))
\end{aligned} \tag{12}$$

Since $L(q(\alpha^t), \phi^t) = \ln p(\bar{z}|(s,a), \phi^t) - KL(q(\alpha^t)||p(\alpha|\bar{z}, (s,a), \phi^t))$ is the lower bound $ll(\phi^t, \alpha^t)$. We update α^t by letting $q(\alpha^t) = p(\alpha|\bar{z}, (s,a), \phi^t)$ to maximize the lower bound, where $p(\alpha|\bar{z}, (s,a), \phi^t)$ can be calculated based on the second equation in Eq. 5. Thus, $KL(q(\alpha^{t+1})||p(\alpha|\bar{z}, (s,a), \phi^t)) = 0$. Thus, $ll(\phi^t, \alpha^{t+1})$ can be expressed as

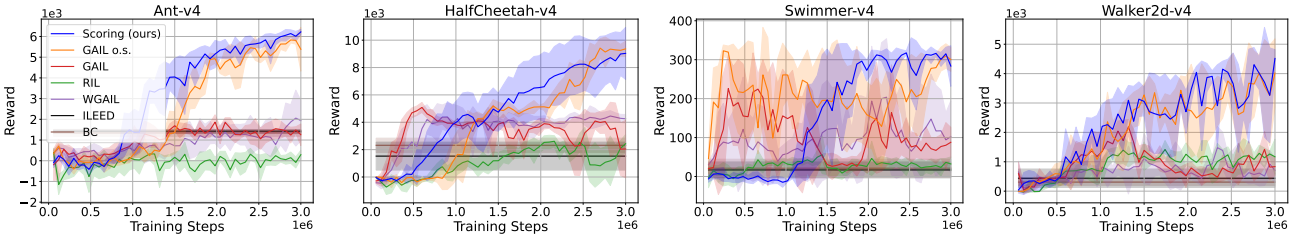
$$\begin{aligned}
ll(\phi^t, \alpha^{t+1}) &= \ln p(\bar{z}|(s,a), \phi^t) \\
&= L(q(\alpha^{t+1}), \phi^t) + KL(q(\alpha^{t+1})||p(\alpha|\bar{z}, (s,a), \phi^t)) \\
&= L(q(\alpha^{t+1}), \phi^t)
\end{aligned} \tag{13}$$

By updating α, we increase the lower bound of $ll(\phi^t, \alpha^{t+1})$ equals to $\ln p(\bar{z}|(s,a), \phi^t)$. Then we maximize $ll(\phi^t, \alpha^{t+1})$ with regard to ϕ using a gradient decent method, $ll(\phi^{t+1}, \alpha^{t+1})$ must be not smaller than $ll(\phi^t, \alpha^{t+1})$. Finally, we have $ll(\phi^{t+1}, \alpha^{t+1}) \geq ll(\phi^t, \alpha^{t+1}) = \ln p(\bar{z}|(s,a), \phi^t) = ll(\phi^t, \alpha^t)$

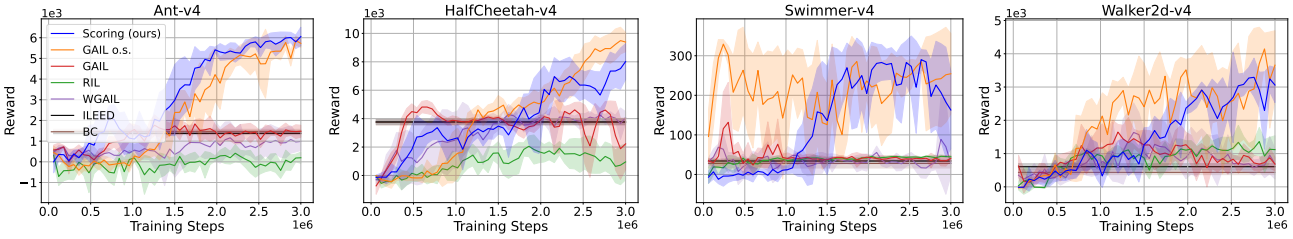
□

E. Experiment Details

E.1. Detailed results & Ablation studies



(a) General expertise test



(b) Low expertise test

Figure 3. Performance comparison (mean $\pm$ std, shown as shaded region) with baselines. Supplementary results for Sec. 4.2.

Table 5. Performance (reward) of checkpoints used as optimal and suboptimal policies across environments. Values are reported as mean $\pm$ standard deviation.

Environment	Optimal policy π^*	Suboptimal policy π^{sub}
Ant-v4	6766.26 ± 109.70	1266.37 ± 533.96
HalfCheetah-v4	15947.53 ± 39.08	4030.23 ± 169.82
Swimmer-v4	339.80 ± 1.15	41.60 ± 2.34
Walker-v4	5638.94 ± 49.76	554.38 ± 267.99

Table 6. True expertise levels $\{\alpha_i\}_{i=1}^m$ for different numbers of demonstrators across two tests. Expertise levels are uniformly distributed within each range.

Demonstrator	1	2	3	4	5	6	7	8
General expertise test								
2 demonstrators	0.100	0.900	–	–	–	–	–	–
4 demonstrators	0.100	0.366	0.633	0.900	–	–	–	–
6 demonstrators	0.100	0.260	0.420	0.580	0.740	0.900	–	–
8 demonstrators	0.100	0.214	0.329	0.443	0.557	0.671	0.786	0.900
Low expertise test								
2 demonstrators	0.050	0.150	–	–	–	–	–	–
4 demonstrators	0.050	0.083	0.117	0.150	–	–	–	–
6 demonstrators	0.050	0.070	0.090	0.110	0.130	0.150	–	–
8 demonstrators	0.050	0.064	0.079	0.093	0.107	0.121	0.136	0.150

Number of Demonstrators: The results in Tab. 11 show that the estimation error of expertise levels in low expertise test increase when the number of demonstrators is reduced. However, the reward obtained by the IL agent in Stage 2 (see Tab. 12) does not decrease substantially. These results demonstrate the robustness of our method with respect to varying demonstrator counts in terms of IL performance.

Table 7. Estimated expertise levels $\{\hat{\alpha}_i\}_{i=1}^m$, showing the mean estimate and the mean $\pm$ standard deviation of error across five random seeds. True values are shown separately. Supplementary results for Sec. 4.3

Demonstrator	1	2	3	4	5	6	Mean error $\pm$ std
General expertise test							
Ant-v4	0.100	0.260	0.420	0.580	0.740	0.900	$(2.67 \pm 1.28) \times 10^{-4}$
HalfCheetah-v4	0.100	0.260	0.420	0.580	0.740	0.900	$(2.00 \pm 2.00) \times 10^{-5}$
Hopper-v4	0.101	0.259	0.418	0.575	0.734	0.892	$(3.83 \pm 2.73) \times 10^{-3}$
Swimmer-v4	0.100	0.260	0.420	0.580	0.740	0.900	$(2.00 \pm 2.00) \times 10^{-5}$
Walker2d-v4	0.100	0.260	0.420	0.581	0.741	0.900	$(3.27 \pm 3.12) \times 10^{-4}$
True expertise levels	0.100	0.260	0.420	0.580	0.740	0.900	
Low expertise test							
Ant-v4	0.148	0.170	0.187	0.207	0.222	0.246	$(9.65 \pm 0.24) \times 10^{-2}$
HalfCheetah-v4	0.086	0.105	0.127	0.146	0.167	0.190	$(3.67 \pm 0.15) \times 10^{-2}$
Hopper-v4	0.156	0.176	0.195	0.216	0.231	0.257	$(1.05 \pm 0.02) \times 10^{-1}$
Swimmer-v4	0.077	0.100	0.118	0.139	0.158	0.177	$(2.81 \pm 0.11) \times 10^{-2}$
Walker2d-v4	0.179	0.206	0.219	0.248	0.263	0.287	$(1.34 \pm 0.04) \times 10^{-1}$
True expertise levels	0.050	0.070	0.090	0.110	0.130	0.150	

Table 8. Classification / labeling results (mean $\pm$ std). Stage 1: Labeling using the pretrained scoring model. Stage 2: Relabeling using the fine-tuned scoring model. TP (true positive), TN (true negative), FP (false positive), FN (false negative). Improvement (%) shows the Stage 1 $\rightarrow$ Stage 2 percentage change of accuracy and precision after relabeling. Supplementary results for Sec. 4.4

Environment	Stage	TP	TN	FP	FN	Accuracy / Precision	Improvement (%)
General expertise test							
Ant-v4	1	14988.6 $\pm$ 3.9	14984.2 $\pm$ 10.1	15.8 $\pm$ 10.1	11.4 $\pm$ 3.9	0.9991 / 0.9989	-
	2	14984.0 $\pm$ 5.5	14984.0 $\pm$ 5.5	16.0 $\pm$ 5.5	16.0 $\pm$ 5.5	0.9989 / 0.9989	-0.02 / 0.00
HalfCheetah-v4	1	15000.0 $\pm$ 0.0	14999.4 $\pm$ 0.8	0.6 $\pm$ 0.8	0.0 $\pm$ 0.0	1.0000 / 1.0000	-
	2	15000.0 $\pm$ 0.0	15000.0 $\pm$ 0.0	0.0 $\pm$ 0.0	0.0 $\pm$ 0.0	1.0000 / 1.0000	0.00 / 0.00
Swimmer-v4	1	15000.0 $\pm$ 0.0	14999.4 $\pm$ 1.2	0.6 $\pm$ 1.2	0.0 $\pm$ 0.0	1.0000 / 1.0000	-
	2	14991.4 $\pm$ 16.2	14991.4 $\pm$ 16.2	8.6 $\pm$ 16.2	8.6 $\pm$ 16.2	0.9994 / 0.9994	-0.06 / -0.06
Walker2d-v4	1	14992.0 $\pm$ 2.2	14985.0 $\pm$ 6.1	15.0 $\pm$ 6.1	8.0 $\pm$ 2.2	0.9992 / 0.9990	-
	2	14990.4 $\pm$ 1.6	14990.4 $\pm$ 1.6	9.6 $\pm$ 1.6	9.6 $\pm$ 1.6	0.9994 / 0.9994	0.02 / 0.04
Low expertise test							
Ant-v4	1	2997.6 $\pm$ 1.5	24101.4 $\pm$ 735.5	2898.6 $\pm$ 735.5	2.4 $\pm$ 1.5	0.9033 / 0.5178	-
	2	2947.0 $\pm$ 32.9	26947.0 $\pm$ 32.9	53.0 $\pm$ 32.9	53.0 $\pm$ 32.9	0.9965 / 0.9823	10.32 / 89.71
HalfCheetah-v4	1	3000.0 $\pm$ 0.0	25898.2 $\pm$ 355.8	1101.8 $\pm$ 355.8	0.0 $\pm$ 0.0	0.9633 / 0.7368	-
	2	2997.4 $\pm$ 5.2	26997.4 $\pm$ 5.2	2.6 $\pm$ 5.2	2.6 $\pm$ 5.2	0.9998 / 0.9991	3.79 / 35.60
Swimmer-v4	1	3000.0 $\pm$ 0.0	26156.2 $\pm$ 393.5	843.8 $\pm$ 393.5	0.0 $\pm$ 0.0	0.9719 / 0.7884	-
	2	2637.2 $\pm$ 290.4	26637.2 $\pm$ 290.4	362.8 $\pm$ 290.4	362.8 $\pm$ 290.4	0.9758 / 0.8791	0.40 / 11.50
Walker2d-v4	1	2965.0 $\pm$ 42.7	22953.2 $\pm$ 890.1	4046.8 $\pm$ 890.1	35.0 $\pm$ 42.7	0.8639 / 0.4301	-
	2	2772.0 $\pm$ 170.3	26772.0 $\pm$ 170.3	228.0 $\pm$ 170.3	228.0 $\pm$ 170.3	0.9848 / 0.9240	13.99 / 114.83

Table 9. Ablation results on the impact of relabeling and early stopping in Stage 2 on Ant-v4 task. Reported as mean $\pm$ std of reward across random seeds; each seed's value is the mean over its last five checkpoints. Supplementary results for Sec. 4.5

Method	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
LEOS-IL (ours)	**6084.6** $\pm$ **146.9**	**5932.0** $\pm$ **335.2**
Scoring (no relabel)	5593.0 $\pm$ 564.2	3316.2 $\pm$ 1207.8
Scoring (no early stop)	2129.8 $\pm$ 2205.8	1018.2 $\pm$ 250.0

Number of Human Queries: To study the sensitivity to the number of human queries, we conducted ablation experiments, varying the number of queries. The results are shown in Tab. 13-14, Fig. 6. We observe that even with only a single query, our method still achieves similar performance to that obtained with 5 queries. This is because the classification accuracy in

Table 10. Ablation study on top-k hyperparameter on Ant-v4 task. Reported as mean ± std of reward across random seeds; each seed's value is the mean over its last five checkpoints.. Supplementary results for Sec. 4.5

Top-k fraction	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
0.05	–	5887.6 ± 263.8
0.1	5403.8 ± 378.7	**5932.0 ± 335.2**
0.15	–	3663.4 ± 1950.9
0.2	–	1756.6 ± 902.6
0.3	5809.0 ± 274.8	–
0.5	**6084.6 ± 146.9**	–
0.55	4109.2 ± 2204.9	–
0.6	3641.0 ± 1237.7	–
0.7	2383.0 ± 1059.7	–

Table 11. Estimated expertise levels $\{\hat{\alpha}_i\}_{i=1}^m$ on Ant-v4 task, showing the mean estimate and the mean ± standard deviation of error across five random seeds. True values are shown separately. Results shown for different numbers of demonstrators (2, 4, 6, 8), each provides 5,000 samples. Supplementary results for Sec. 4.5

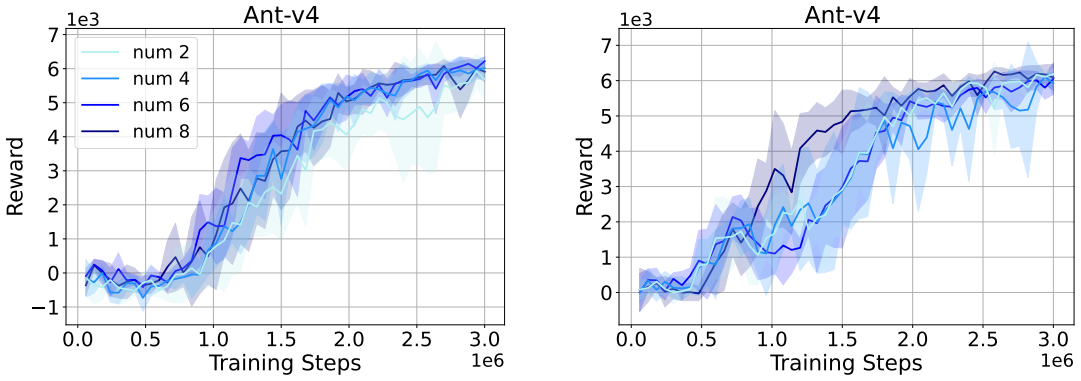
Demonstrator	Type	1	2	3	4	5	6	7	8	Mean error ± std
General expertise test										
2 demonstrators	Est.	0.101	0.899	-	-	-	-	-	-	$(9.60 \pm 1.60) \times 10^{-4}$
	True	0.100	0.900	-	-	-	-	-	-	-
4 demonstrators	Est.	0.100	0.366	0.633	0.899	-	-	-	-	$(4.50 \pm 2.96) \times 10^{-4}$
	True	0.100	0.367	0.633	0.900	-	-	-	-	-
6 demonstrators	Est.	0.100	0.260	0.420	0.580	0.740	0.900	-	-	$(2.67 \pm 1.28) \times 10^{-4}$
	True	0.100	0.260	0.420	0.580	0.740	0.900	-	-	-
8 demonstrators	Est.	0.100	0.214	0.329	0.443	0.557	0.671	0.785	0.899	$(2.50 \pm 1.97) \times 10^{-4}$
	True	0.100	0.214	0.329	0.443	0.557	0.671	0.786	0.900	-
Low expertise test										
2 demonstrators	Est.	0.193	0.287	-	-	-	-	-	-	$(1.40 \pm 0.03) \times 10^{-1}$
	True	0.050	0.150	-	-	-	-	-	-	-
4 demonstrators	Est.	0.106	0.143	0.173	0.208	-	-	-	-	$(5.77 \pm 0.16) \times 10^{-2}$
	True	0.050	0.083	0.117	0.150	-	-	-	-	-
6 demonstrators	Est.	0.148	0.170	0.187	0.207	0.222	0.246	-	-	$(9.65 \pm 0.24) \times 10^{-2}$
	True	0.050	0.070	0.090	0.110	0.130	0.150	-	-	-
8 demonstrators	Est.	0.073	0.087	0.101	0.115	0.130	0.144	0.160	0.175	$(2.31 \pm 0.10) \times 10^{-2}$
	True	0.050	0.064	0.079	0.093	0.107	0.121	0.136	0.150	-

Table 12. Ablation on the different number of demonstrators (each providing 5,000 samples) for Ant-v4 task. Reported numbers are mean ± std across random seeds; each seed's value is the mean over its last five checkpoints. Supplementary results for Sec. 4.5

Number of Demonstrators	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
2	5634.0 ± 617.8	6050.6 ± 120.4
4	5934.0 ± 264.3	5637.2 ± 772.5
6	**6084.6 ± 146.9**	5932.0 ± 335.2
8	5735.0 ± 225.6	**6146.6 ± 107.2**

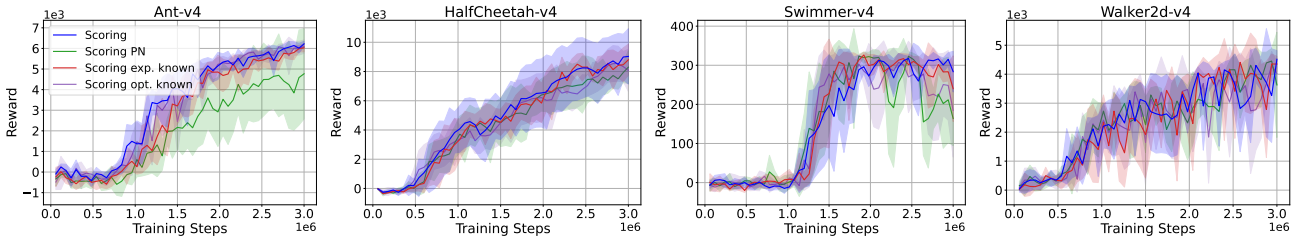
Stage 1 is extremely high (above 99%), making it very unlikely for the system to select an incorrectly labeled sample that would lead to completely opposite or flipped classification results. Also, the learned policies across different numbers of queries have similar performance in final reward achieved.

Then, we explain the necessity of this query. In our problem setting, we do not have labels for either sample optimality or expertise levels. Without any labels, our classification method effectively performs an operation similar to clustering on the unlabeled data. A limited number of human queries is then used to ensure that the resulting clusters are assigned the correct

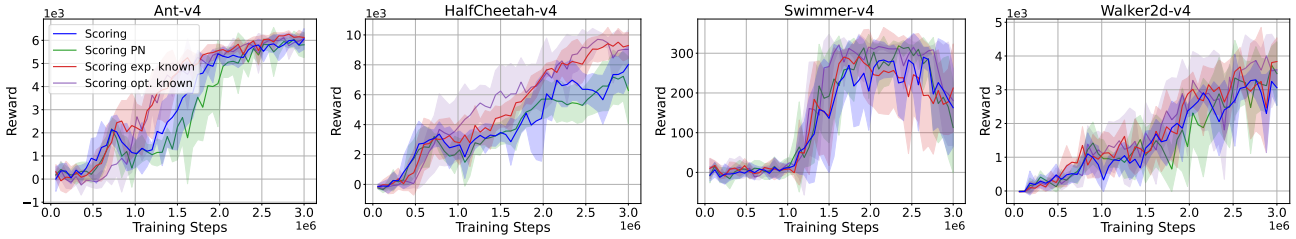


(a) Expertise level range [0.1 - 0.9] *(b)* Expertise level range [0.05 - 0.15]

Figure 4. Ablation on different numbers of demonstrators for Ant-v4 task (mean $\pm$ std, shown as shaded region). Supplementary results for Sec. 4.5



(a) General expertise test



(b) Low expertise test

Figure 5. Comparative analysis of our method and its variants.

labels. We select 5 as the number of queries so that the procedure remains both effective and practical.

Expected Expertise α': To study the sensitivity to the value of α', we conduct ablation studies examining the effect of different α' values. Tab. 15 presents the classification results of the scoring model, while Tab. 16 and Fig. 7 report the rewards obtained by the learned policies. The performance of both the scoring model and the learned policy is maximized when α' is close to 0.5, and degrades when α' is extremely small or large. These results indicate that selecting $\alpha' = 0.5$ yields the best performance for both the general expertise and low expertise test settings.

Number of Initializations for Scoring Model Training: We report the number of initializations required in Alg. 1 in Tab. 17. For the general expertise test, we use a single initialization until the Expertise Dispersion Criterion is satisfied. In more challenging low expertise cases, the number of initializations increases but remains fewer than 10. These results demonstrate that our algorithm is computationally efficient in practice.

E.2. Additional Analysis of Method Variants

We analyze our algorithm by comparing it against several variants under different settings. (1) **Scoring**: original method; (2) **Scoring PN**: in Stage 2, the first term $\widehat{\mathbb{E}}_{(s,a)\sim\rho}\big[\mathcal{L}(f_\phi(s,a)\big]$ (demonstration discriminative loss) in Eq. 6 is re-

Table 13. Classification results of the scoring model on imperfect demonstrations (Ant-v4) for Stage 1 and Stage 2 (Accuracy / Precision, %), with Stage 1 → Stage 2 relative improvement (%). Ablation study about expert query count. Supplementary results see Fig. 6

Test	Stage	Query 1	Impr.	Query 3	Impr.	Query 5	Impr.	Query 7	Impr.	Query 9	Impr.
General	1	99.93 / 99.97	-	99.92 / 99.94	-	99.91 / 99.89	-	99.92 / 99.96	-	99.94 / 99.95	-
Expertise	2	99.92 / 99.92	-0.01 / -0.05	99.90 / 99.90	-0.02 / -0.04	99.89 / 99.89	-0.02 / 0.00	99.89 / 99.89	-0.03 / -0.07	99.91 / 99.91	-0.03 / -0.04
Low	1	96.26 / 75.31	-	97.33 / 80.17	-	90.33 / 51.78	-	97.02 / 78.81	-	97.14 / 79.26	-
Expertise	2	99.64 / 98.22	3.51 / 30.42	99.52 / 97.60	2.25 / 21.74	99.65 / 98.23	10.32 / 89.71	99.48 / 97.40	2.54 / 23.59	99.73 / 98.65	2.67 / 24.46

Table 14. Ablation study on Query times for Ant-v4. Reported as mean ± std of reward across random seeds; each seed's value is the mean over its last five checkpoints.

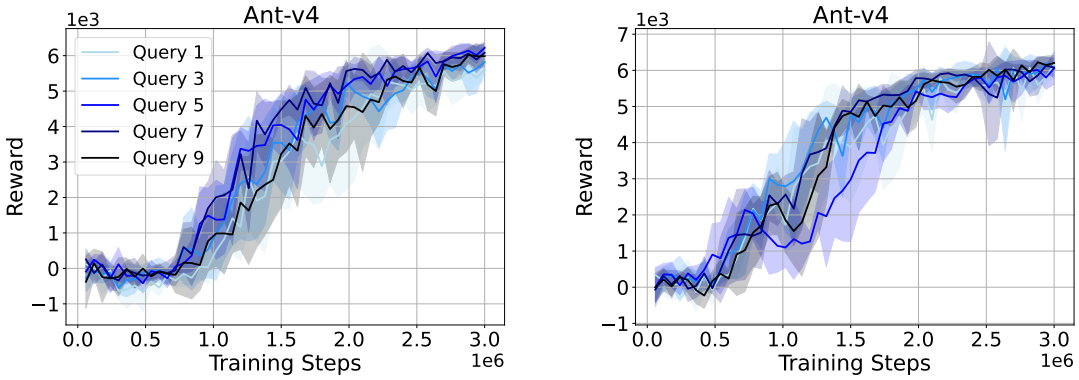
Query times	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
1	5433.6 ± 365.6	5810.2 ± 301.3
3	5655.2 ± 307.3	6011.2 ± 219.9
5	6084.6 ± 146.9	5932.0 ± 335.2
7	5981.4 ± 264.6	5944.2 ± 279.8
9	5881.6 ± 265.3	6130.8 ± 126.8

placed with a positive-negative (PN) classification loss (binary cross-entropy) $\widehat{\mathbb{E}}_{\substack{(s,a)\sim\hat{\rho}^{\pi^*} \\ (s',a')\sim\hat{\rho}^{\pi^{\text{sub}}}}} \left[\mathcal{L}_{PN}(f_\phi(s,a,s',a')\right]$, where $\mathcal{L}_{PN}(f_\phi(s,a,s',a') = \log\big(1 - f_\phi(s,a)\big) + \log\big(f_\phi(s',a')\big)$, and $\hat{\rho}^{\pi^{\text{sub}}}$ represents the pseudo-suboptimal data distribution inferred from $D_{\text{subopt}} = D \setminus D_{\text{opt}}$. (3) **Scoring exp. known**: the true expertise levels are assumed to be given in both Stage 1 and 2. (4) **Scoring opt. known**: in Stage 2, D_{opt} is provided based on the true labels.

The results in Fig. 5 show that, overall, all methods achieve comparable performance. **Scoring PN** yields slightly lower rewards in general expertise test on Ant and Swimmer tasks, which may be attributed to the overconfidence issue introduced by self-labeling when using the PN loss. This result highlights the effectiveness of our demonstration discriminative loss. **Scoring exp. known** and **Scoring opt. known** serve as oracle variants (with oracle knowledge of expertise levels and optimality labels, respectively) and achieve only marginally better performance than the original method in the low expertise test on HalfCheetah and Walker2d. This observation highlights the effectiveness of our approach in expertise-level estimation and optimality prediction when learning from imperfect demonstrations.

E.3. Reproducibility Statement

To ensure the reproducibility of this work, details of the training parameters are provided in Tab. 19. All reported results are averaged over five random seeds (0–4) per condition/method, with mean and standard deviation reported. The corresponding code is publicly available at: `https://github.com/ch-engpan/IL_imperfect`



(a) Expertise level range [0.1 - 0.9]

(b) Expertise level range [0.05 - 0.15]

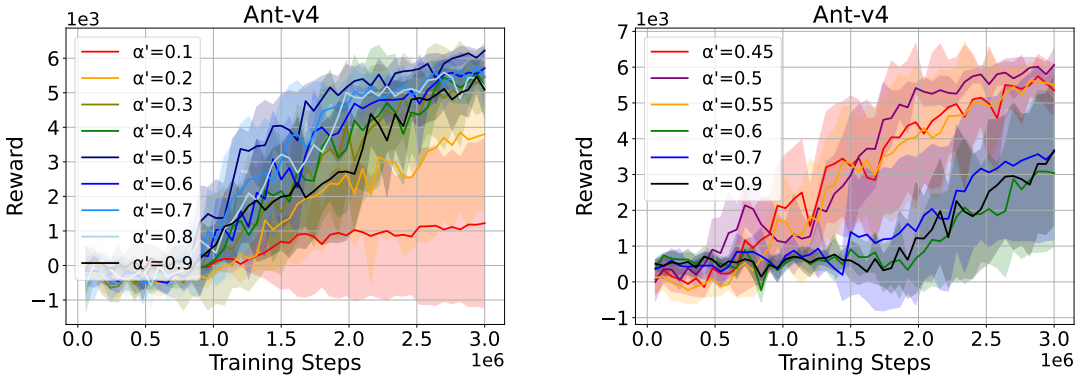
Figure 6. Ablation on the different expert query count for Ant-v4 task (mean ± std, shown as shaded region). Supplementary results see Tab. 13

Table 15. Classification results of the scoring model on imperfect demonstrations for Stage 1 - Accuracy and Precision (%, mean ± std) - for Ant-v4 across different α' values (expected expertise in the target environment). Supplementary results see Fig. 7

	General expertise level		**Low expertise level**	
α'	**Accuracy**	**Precision**	**Accuracy**	**Precision**
0.1	59.97 ± 19.94	20.00 ± 39.99	90.00 ± 0.00	0.00 ± 0.00
0.2	89.94 ± 19.97	79.97 ± 39.99	90.00 ± 0.00	0.00 ± 0.00
0.3	99.92 ± 0.02	99.96 ± 0.03	-	-
0.4	99.94 ± 0.01	99.98 ± 0.02	90.00 ± 0.00	0.00 ± 0.00
0.45	-	-	96.44 ± 2.43	88.46 ± 11.92
0.5	99.91 ± 0.03	99.89 ± 0.07	90.33 ± 2.45	51.78 ± 7.56
0.55	-	-	81.07 ± 35.54	77.18 ± 34.12
0.6	99.92 ± 0.02	99.90 ± 0.05	20.03 ± 20.07	12.01 ± 4.02
0.7	99.93 ± 0.02	99.93 ± 0.05	27.21 ± 34.42	22.35 ± 12.70
0.8	99.92 ± 0.02	99.90 ± 0.04	-	-
0.9	79.91 ± 24.43	79.87 ± 24.39	10.00 ± 0.00	10.00 ± 0.00

Table 16. Ablation study on α' for Ant-v4. Reported as mean ± std of reward across random seeds; each seed's value is the mean over its last five checkpoints. Note that there are no policy learning results appear for expertise [0.05 – 0.15] when $\alpha' \leq 0.4$ because all imperfect demonstrations are labeled sub-optimal, leaving the optimal set empty.

α' value	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
0.1	1149.4 ± 2298.8	–
0.2	3700.6 ± 2286.5	–
0.3	5241.6 ± 957.9	–
0.4	5308.0 ± 377.7	–
0.45	–	5509.0 ± 493.4
0.5	6084.6 ± 146.9	5932.0 ± 335.2
0.55	–	5496.6 ± 509.1
0.6	5579.0 ± 416.2	2927.8 ± 2064.2
0.7	5542.8 ± 261.6	3497.6 ± 2349.8
0.8	5262.6 ± 330.8	–
0.9	5090.2 ± 573.1	3171.0 ± 1910.6



(a) Expertise level range [0.1 - 0.9]

(b) Expertise level range [0.05 - 0.15]

Figure 7. Ablation on the different α' values (expected expertise in the target environment) for Ant-v4 task (mean $\pm$ std, shown as shaded region). Supplementary results see Tab. 15

Env	General expertise (Mean ± Std)	Low expertise (Mean ± Std)
Ant-v4	1.00 ± 0.00	8.80 ± 3.31
HalfCheetah-v4	1.00 ± 0.00	9.80 ± 1.94
Swimmer-v4	1.00 ± 0.00	3.60 ± 4.72
Walker2d-v4	1.00 ± 0.00	2.80 ± 3.12

Table 17. Number of repeated computation loops required in Alg. 1 (Mean ± Std), using random initializations until the Expertise Dispersion Criterion are satisfied

Table 18. Performance comparison of BC, ILEED, and LEOS-IL (ours), reported as mean $\pm$ std of reward. The "best policy" refers to the checkpoint with the highest true reward (oracle) over the entire training process, while "last 5 policies mean" denotes the average performance over the final five training checkpoints.

Environment	BC (best policy)	ILEED (best policy)	BC (last 5 policies mean)	ILEED (last 5 policies mean)	LEOS-IL (ours), (last 5 policies mean)
General expertise test					
Ant-v4	2088.5 ± 42.0	2188.6 ± 70.3	1341.8 ± 164.5	1428.2 ± 80.3	**6084.6 ± 146.9**
HalfCheetah-v4	3352.5 ± 214.2	3545.7 ± 173.0	2318.1 ± 521.2	1526.2 ± 1020.4	**8751.2 ± 1797.9**
Swimmer-v4	329.4 ± 17.7	344.6 ± 1.4	21.3 ± 23.0	16.8 ± 20.5	**305.4 ± 15.7**
Walker2d-v4	1472.7 ± 127.9	1439.2 ± 97.2	308.4 ± 111.5	440.3 ± 313.0	**3959.2 ± 671.4**
Low expertise test					
Ant-v4	2033.2 ± 55.7	2041.3 ± 32.6	1473.4 ± 97.7	1376.5 ± 62.1	**5932.0 ± 335.2**
HalfCheetah-v4	4077.1 ± 24.9	4074.9 ± 33.3	3753.9 ± 220.5	3767.6 ± 176.9	**7335.0 ± 1316.4**
Swimmer-v4	220.3 ± 71.3	255.2 ± 54.7	27.8 ± 11.6	34.0 ± 2.4	**195.4 ± 92.1**
Walker2d-v4	1345.3 ± 110.0	1509.3 ± 213.9	436.6 ± 69.1	605.3 ± 94.5	**2961.8 ± 339.8**

Table 19. Training Parameters

Stage	Parameter	Value
Stage 1	f_ϕ learning rate	1×10^{-4}
	f_ϕ batch size	1000
	f_ϕ optimizer	Adam
	Variance threshold ϵ for expertise levels	0.0005
	Expertise levels estimation frequency	Every 5 epochs
	f_ϕ training epochs	50 (Ant, HalfCheetah, Swimmer) 100 (Walker2d)
Stage 2	π_θ model	SAC (Stable-Baselines3)
	π_θ learning starts from	100 steps
	π_θ batch size	1024
	π_θ learning rate	2×10^{-3}
	π_θ replay buffer size	1×10^{6}
	π_θ training steps / IL iteration	1.5×10^{4}
	f_ϕ learning rate	1×10^{-4}
	f_ϕ batch size	1024
	f_ϕ optimizer	Adam
	f_ϕ updates epochs / IL iteration	500
	Δ_t, p for relabeling early stopping	30 (for 5 consecutive steps)
	IL total iterations	200
	checkpoints saving freqency	1 per IL iteration