FROM MANY IMPERFECT TO ONE TRUSTED: IMITATION LEARNING FROM HETEROGENEOUS DEMONSTRATORS WITH UNKNOWN EXPERTISE

Anonymous authors
Paper under double-blind review

ABSTRACT

Imitation learning (IL) typically depends on large-scale demonstrations collected from *multiple* human or algorithmic demonstrators. Yet, most existing methods assume these demonstrators are either homogeneous or near-optimal—a convenient but unrealistic assumption in many real-world settings. In this work, we tackle a more practical and challenging setting: IL from *heterogeneous* demonstrators with *unknown* and *widely varying expertise levels*. Instead of assuming expert dominance, we model each demonstrator's behavior as a *flexible* mixture of optimal and suboptimal policies, and propose a novel IL framework that jointly learns (a) a state-action optimality scoring model and (b) the latent expertise level of each demonstrator, using only a handful of human queries. The learned scoring model is then integrated into an policy optimization procedure, where it is fine-tuned with offline demonstrations, on-policy rollouts, and a fine-grained mixup regularizer to produce informative rewards. The agent is trained to maximize these learned rewards in an iterative fashion. Experiments on continuous-control benchmarks show that our approach consistently outperforms baseline methods. Even when all demonstrators are *highly suboptimal*, each exhibiting only $5-15\%$ optimality, our method achieves performance comparable to a baseline trained on purely optimal demonstrations, despite our lack of optimality labels.

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 demonstrators whose expertise levels are unknown and variable, naturally leading to datasets containing suboptimal demonstrations. 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 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 any of these assumptions?

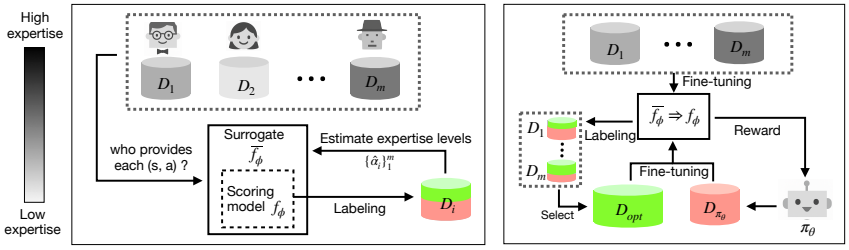


Figure 1: **Overview of our heterogeneous IL framework**, illustrated with m demonstrators possessing (unknown) heterogeneous expertise levels (represented in grayscale; darker shades indicate higher expertise). Green/red indicate inferred optimal/suboptimal pairs. **Stage 1** (left panel): Joint learning 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) $\{\alpha_i\}_{i=1}^m$ 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}.

This problem is inherently challenging because neither demonstrator expertise levels nor demonstration optimality is observed. In the worst case, all demonstrators may exhibit very low expertise, making it extremely difficult to extract reliable signals for policy learning.

Our first contribution is a novel problem formulation for *IL from heterogeneous demonstrators*, where each demonstrator's policy is modeled as a mixture of optimal and suboptimal behaviors, and their *expertise level* is characterized as the proportion of their actions that align with the optimal policy. These expertise levels *vary* across demonstrators, capturing their heterogeneity, but are latent and unknown, leaving little to no direct information about the quality of the demonstrations. Our goal is to learn a high-quality policy from this heterogeneous and suboptimal dataset.

Our second contribution is a novel two-stage methodology for addressing the above problem (Figure 1). In the first stage, we jointly estimate demonstrator expertise levels and learn a *scoring model* that assigns *optimality scores* to the state-action pairs, representing the likelihood that the pair corresponds to the (latent) optimal policy. This is achieved through an EM-style iterative procedure, and we also provide a theoretical analysis showing that this process converges under mild assumptions. In the second stage, the learned scoring model serves as a *surrogate reward function* for policy learning. However, since it is trained entirely offline, it may suffer from covariate shift during policy rollouts. To mitigate this, we introduce a refinement procedure that alternates between policy learning and scoring model updates. This feedback loop produces a more informative reward signal, which is then used to further improve the agent policy.

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) (Bain & Sammut, 1995), as well as more recent methods like *generative adversarial imitation learning* (GAIL) (Ho & Ermon, 2016), which matches the state–action occupancy measures of the learner and the demonstrator via adversarial training. Even methods designed for noisy demonstrations, such as *robust imitation learning* (RIL) (Tangkaratt et al., 2020), retain this homogeneity assumption by modeling demonstrators' behaviors as generated from a fixed mixture of optimal and suboptimal policies with a constant mixing coefficient representing expertise level, and assume expert data dominance. While effective in controlled benchmarks, these methods may fail in more realistic multi-demonstrator scenarios where expertise levels vary widely.

Table 1: Comparison of IL methods. ✓ = satisfies criterion, ✗ = does not. "Non-dominant opt. 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 subopt." indicates whether the method can manage structured (i.e., non-random) suboptimal demonstrations. "No opt. labels needed" indicates no labels about the optimality of demonstrations is needed. "Prob. demo scoring" denotes whether the method infers a probabilistic measure of demonstration quality.

Method	BC	GAIL	RIL	IC-GAIL	PU-GAIL	RCE	WGAIL	ILEED	Ours
Non-dominant opt. demo	✗	✗	✗	✓	✓	✗	✗	✓	✓
Multi-demo support	✗	✗	✗	✗	✗	✗	✓	✓	✓
Handles structured subopt.	✗	✗	✓	✓	✓	✗	✓	✗	✓
No opt. labels needed	✗	✗	✓	✗	✓	✗	✓	✓	✓
Prob. demo scoring	✗	✓	✓	✗	✓	✓	✗	✗	✓

IL with Optimality Labels To better handle suboptimal demonstrations, many IL methods use supervision to distinguish optimal from non-optimal behavior. This strategy underlies extensions of GAIL. For example, *imperfect demonstration and confidence GAIL* (IC-GAIL) (Wu et al., 2019) uses confidence labels to reweight imperfect demonstrations, and *positive-unlabeled GAIL* (PU-GAIL) (Wang et al., 2023) leverages positively labeled success trajectories. 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 work well when labeled data is available but are unsuitable when such supervision is hard to obtain.

IL without Homogeneity or Labels More recent efforts aim to learn from heterogeneous, unlabeled, and suboptimal demonstrations without relying on any of the above assumptions. *Weighted GAIL* (WGAIL) (Wang et al., 2021) proposes to estimate importance weights directly from discriminator outputs, enabling the policy to emphasize near-optimal samples without requiring prior knowledge of demonstrator quality. *Imitation learning by estimating expertise of demonstrators* (ILEED) (Beliaev et al., 2022) addresses the same challenge by jointly learning a policy and state-dependent expertise levels, treating expertise as a latent variable conditioned on state and demonstrator embeddings and optimized via maximum likelihood. However, their formulation assumes that suboptimal demonstrations arise from a uniformly random distribution, which limits its ability to handle structured suboptimality. Building on this direction, our work defines expertise level as the prior probability that a state–action pair comes from an optimal policy. This formulation accommodates structured suboptimality and eases estimation by avoiding strong state dependence.

3 Imitation Learning from Heterogeneous Demonstrators

3.1 Problem Setup

In reinforcement learning (RL), 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 whose induced state-action distribution matches that of the optimal policy (Ziebart et al., 2010).

To make this feasible, most IL algorithms assume that *all* demonstrations are generated by the optimal policy. However, this assumption is overly restrictive, as real datasets are often collected from multiple demonstrators with varying skills. In this work, we study a more practical IL setting where demonstrations are provided by heterogeneous demonstrators with different expertise levels.

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 that demonstrator i tends to act more consistently with π^*. We consider *demonstrator heterogeneity*, i.e., there $\exists\, i, j \in [m], i \neq j$, 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*.

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.

Since demonstrators are heterogeneous, we model the policy of each demonstrator-i as a mixture

$$\pi_i(a \mid s) = \alpha_i \, \pi^*(a \mid s) \; + \; (1 - \alpha_i) \, \pi^{\text{sub}}(a \mid s). \tag{1}$$

Let $\rho^{\pi_i}(s)$ be the state visitation distribution of π_i. The occupancy measure for demonstrator-i is

$$\rho_i(s, a) = \rho^{\pi_i}(s) \; \pi_i(a \mid s) = \rho^{\pi_i}(s) \left(\alpha_i \, \pi^*(a \mid s) + (1 - \alpha_i) \pi^{\text{sub}}(a \mid s) \right). \tag{2}$$

Similarly, the conditional occupancy measures of the optimal and suboptimal policies are

$$P(s, a \mid z = 1) = \rho^{\pi^*}(s) \, \pi^*(a \mid s), \quad P(s, a \mid z = 0) = \rho^{\pi^{\text{sub}}}(s) \, \pi^{\text{sub}}(a \mid s). \tag{3}$$

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* $f_\phi(s, a) \approx \mathbb{P}(z = 1|s, a)$, which estimates the optimality scores and acts as a surrogate reward: $\pi^* = \arg\max_\pi \mathbb{E}_{p_\pi} \left[\sum_{t=1}^{T} \gamma^t f_\phi(s_t, a_t) \right]$.

3.2 Joint estimation of expertise levels 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 solve this problem, we adopt a surrogate learning approach inspired by the surrogage set classification (SSC) framework of Lu et al. (2021).

Surrogate Demonstrator Classification We use the demonstrator identity $\bar{z} \in [m]$ as a surrogate label. Defining $\mathcal{S}$ as the set of states and $\mathcal{A}$ as the set of actions, 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:

$$\mathcal{L}(\overline{f}_\phi) = \mathbb{E}_{(s,a) \sim \sum_i \mu_i \rho_i(s,a)} [\ell(\overline{f}_\phi(s, a), \bar{z})], \tag{4}$$

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: $\overline{f}_\phi = T_i(f_\phi)$,[1] where the i-th component of T is

$$T_i(x) = \frac{\mu_i(\alpha_i - \alpha')x + \mu_i \alpha'(1 - \alpha_i)}{\sum_{i=1}^m \mu_i(\alpha_i - \alpha')x + \sum_{i=1}^m \mu_i \alpha'(1 - \alpha_i)}, \tag{5}$$

Here, $\alpha_i = \mathbb{P}(z = 1|\bar{z} = i)$ is the expertise level of demonstrator i, α' is the expected expertise in the target environment where the scoring model is deployed. Since this expected expertise is unknown, we set α' as 0.5 in this work. If the expertise levels are known, then f_ϕ can be learned by minimizing the following equivalent form of Eq. 4:

$$\mathcal{L}(f_\phi) = \mathbb{E}_{(s,a) \sim \rho} \left[\ell(T(f_\phi(s, a)), \bar{z}) \right]. \tag{6}$$

Once $\overline{f}_\phi$ is trained optimally, the scoring model f_ϕ can be recovered by removing the transformation layer, as established in Lu et al. (2021).

[1] With a slight abuse of notation, we use f_ϕ and $\overline{f}_\phi$ interchangeably 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

Input: Dataset $D = \{(i, D_i)\}_{i=1}^m$, number of sample 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 scoring model):
    Update f_ϕ by minimizing the empirical risk from Eq. 6 using current $\{\alpha_i\}_{i=1}^m$
    E-step (Estimate expertise levels):
    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

3.2.2 EM-STYLE OPTIMIZATION OF SCORING MODEL AND EXPERTISE ESTIMATES

However, neither optimality labels nor expertise levels are observed in our setting. To jointly learn both, we propose an expectation-maximization (EM)-style algorithm, outlined in Algorithm 1.

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. 6. We then 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. Here, we present a convergence result that guarantees the EM-style optimization of of α and ϕ converges to a stationary point. The proof is in Appendix B.

Theorem 1 (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, e^t) be the parameters at iteration t. If the updates follow:*

$$\phi^{t+1} = \arg\min_{\phi,\alpha} \mathcal{L}(\phi^t, \alpha^{t+1}), \quad \alpha^{t+1} = \frac{\sum_{(s,a)\in D_i}[f_{\phi^t}(s,a) > 0.5]}{||D_i||} \tag{7}$$

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

While Theorem 1 guarantees convergence, the quality of the resulting 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 and use the following two criteria to select the best solution.

Expertise Dispersion Criterion Since our algorithm relies on self-labeled data during training, it is susceptible to error propagation and confirmation bias. Empirically, we observe that certain random initializations lead to degenerate solutions, where the scoring model 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 $\mathrm{Var}(\{\alpha_i\}_{i=1}^m)$. If the variance falls below a small threshold $\epsilon > 0$, we consider the model to have failed in distinguishing between different expertise levels, as the predictions carry little information, analogous to having low entropy, and are indicative of overfitting. 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 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:

$$\alpha_i \leftarrow 1 - \alpha_i,\ f_\phi(s,a) \leftarrow 1 - f_\phi(s,a),\ \ \forall i \in [m],\ \forall (s,a) \in D.$$

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

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 from Eq. 8
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

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, which limits its generalization to learner-generated 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* in Eq. 6, 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 overall objective for refining f_ϕ is given by:

$$\min_{f_\phi} \widehat{\mathbb{E}}_{(s,a)\sim\rho}\big[\mathcal{L}(f_\phi(s,a)\big] + \widehat{\mathbb{E}}_{(s',a')\sim\rho^{\pi_\theta}}\big[\log(f_\phi(s',a'))\big] + \widehat{\mathbb{E}}_{\substack{(s,a)\sim\hat{\rho}^{\pi^*}\\(s',a')\sim\rho^{\pi_\theta}}}\big[\mathcal{L}_{\text{mixup}}(s,a,s',a')\big] \quad (8)$$

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

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

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, the mixup regularizer 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 outlined 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. 8 (Line 5), computed over D, D_{π_θ}, and D_{opt}; (4) we relabel D_{opt} using the updated f_ϕ (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. However, these samples are still treated as suboptimal during scoring model refinement. Notably, 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: while GAIL relies exclusively on optimal demonstrations for supervision, our method operates in an (almost) unsupervised setting, allowing f_ϕ to

learn from a large number of suboptimal demonstrations provided by heterogeneous demonstrators. This makes our approach particularly advantageous in scenarios where 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 10). Specifically, we use $\log f_\phi(s, a)$ as the reward signal in reinforcement learning. This policy is updated using soft actor-critic (SAC) (Haarnoja et al., 2018) iteratively, alternating with scoring model refinement and pseudo-label updates.

4 EXPERIMENTS

We will 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 from demonstrators with expertise levels in a normal 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 have mostly focused on cases where most demonstrations are optimal, with only 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 Appendix). We set the top-k hyperparameter k to match the average optimal ratio in each test.

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 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. 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 corresponding state and action spaces of each environment. Moreover, all baselines incorporate mixup regularization. All learner policies are updated using SAC with same hyperparameters. Training details are provided in Tab. 13 in the Appendix. We use 5 random seeds per condition and report the mean and standard deviation in all our results.

The results are summarized in Tab. 2 (see Fig. 3 in the Appendix for training curves). 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 reward in Ant, Swimmer and Walk2d. This is attributed to that our algorithm can achieve

Table 2: Performance comparison with baselines, mean ± std of reward across last five checkpoints

Environment	Dataset	GAIL o.s.	GAIL	RIL	WGAIL	Scoring (ours)
General expertise test						
Ant-v4	4016.3 ± 2776.8	5660.4 ± 163.6	1387.2 ± 301.0	101.8 ± 118.0	1744.0 ± 919.1	**6084.6 ± 146.9**
HalfCheetah-v4	9988.9 ± 5959.9	9278.2 ± 496.5	3035.8 ± 1186.4	1631.6 ± 959.7	4336.2 ± 335.6	**8751.2 ± 1797.9**
Swimmer-v4	190.7 ± 149.1	293.6 ± 28.2	76.6 ± 28.3	28.8 ± 16.3	115.6 ± 83.7	**305.4 ± 15.7**
Walker2d-v4	3096.7 ± 2549.6	3703.0 ± 1011.9	948.6 ± 297.2	1187.2 ± 130.1	665.4 ± 330.2	**3959.2 ± 671.4**
Low expertise test						
Ant-v4	1816.4 ± 1726.3	5542.0 ± 168.1	1439.6 ± 202.0	2.2 ± 174.1	971.6 ± 591.0	**5932.0 ± 335.2**
HalfCheetah-v4	5222.0 ± 3578.8	9194.2 ± 848.1	3162.8 ± 1417.2	964.2 ± 1100.1	3975.6 ± 369.9	**7335.0 ± 1316.4**
Swimmer-v4	71.4 ± 89.5	244.4 ± 113.5	37.4 ± 10.5	44.6 ± 2.9	48.8 ± 53.7	**195.4 ± 92.1**
Walker2d-v4	1062.8 ± 1546.5	3532.6 ± 782.2	737.8 ± 163.2	1140.0 ± 319.0	703.8 ± 291.2	**2961.8 ± 339.8**

Table 3: Estimation errors of Stage 1 scoring model for expertise levels (mean ± 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	$(9.65 \pm 0.24) \times 10^{-2}$	$(3.67 \pm 0.15) \times 10^{-2}$	$(1.05 \pm 0.02) \times 10^{-1}$	$(2.81 \pm 0.11) \times 10^{-2}$	$(1.34 \pm 0.04) \times 10^{-1}$

high classification performance on imperfect demonstrations (see Tab. 4), in contrast to GAIL o.s., is also capable of leveraging suboptimal data and perform better in the subsequent IL stage. In the low xpertise 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 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, which is approximated by assuming that the discriminator in the early stage of GAIL is near-optimal. Although early stopping is used for weight estimation, this assumption can fail due to the instability of adversarial learning.

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 labels on imperfect demonstrations (see Tab. 8). However, these incorrect labels are largely corrected during Stage 2.

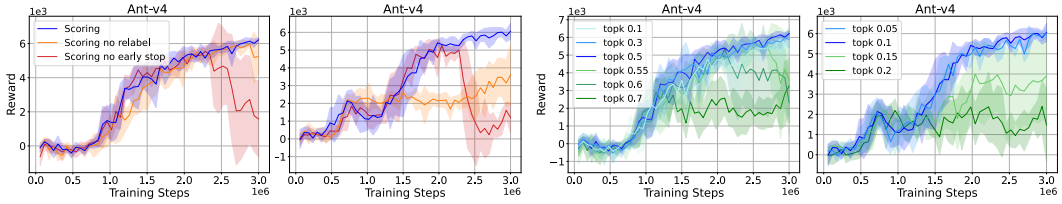
4.4 Optimality Prediction on Imperfection Demonstrations

We evaluate the scoring model f_ϕ on its ability to classify imperfect demonstrations in terms of optimality. Labeling is conducted in two stages: Stage 1 using the pretrained f_ϕ and Stage 2 using the fine-tuned f_ϕ. Performance is measured by accuracy and precision, with emphasis on precision, as it reflects the false positive (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 maintains 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-

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	1	99.91 / 99.89	-	100.00 / 100.00	-	100.00 / 100.00	-	99.92 / 99.90	-
expertise	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	1	90.33 / **51.78**	-	96.33 / **73.68**	-	97.19 / **78.84**	-	86.39 / **43.01**	-
expertise	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

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 (Alg. 2, step 7) and early stopping mechanism (Alg. 2, step 8) in Stage 2, shown in the subfigure (a)-(b) of Fig. 2 (see Tab. 9 for numerical results). Without relabeling, performance is comparable to the original algorithm in the general expertise test but drops by about half in the low expertise test, due to numerous FP errors generated in Stage 1 (see Sec. 4.4). Without the relabeling step in Stage 2, these critical FP errors cannot be corrected using the fine-tuned f_ϕ, causing the performance drop. Without early stopping during relabeling, rewards decline in both general and low expertise tests as the agent approaches the optimal policy. This occurs because f_ϕ can be "fooled" when the agent generates near-optimal samples, increasing classification errors and accumulating incorrect labels, which reinforces overconfidence of f_ϕ and corrupts the surrogate reward for the agent.

Top-k k Value Selection: We study how different k values could influence the final performance of IL. Results in subfigure (c)-(d) in Fig. 2 show that k can be treated as a hyperparameter without prior knowledge of the distribution of expertise (numerical results in Tab. 10). For both the general and low expertise tests, where the average optimality ratio (defined as k^*) of demonstrations is 0.5 and 0.1 respectively, when $k < k^*$ rewards remain comparable to that when $k = k^*$. However, the performance declines as k increases when $k > k^*$. 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 conduct an ablation study on the number of demonstrators (each providing 5,000 samples). Results shows that our methods is robust to varying demonstrator counts in terms of IL performance (see Fig. 4 and Tab. 11- 12 in the Appendix for details).

Appendix C.2 compares our method with variants (e.g., oracle settings, PN loss), showing it consistently matches or outperforms them, demonstrating its robustness and effectiveness.

5 Conclusions

We presented a general IL framework for handling unlabeled, imperfect demonstrations from heterogeneous 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-quality demonstration regimes, highlighting the value of leveraging suboptimal data in an (almost) unsupervised manner.

REFERENCES

Michael Bain and Claude Sammut. A framework for behavioural cloning. In *Machine Intelligence 15*, pp. 103–129, 1995.

Mark Beliaev, Andy Shih, Stefano Ermon, Dorsa Sadigh, and Ramtin Pedarsani. Imitation learning by estimating expertise of demonstrators. In *International Conference on Machine Learning*, pp. 1732–1748. PMLR, 2022.

Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. *Advances in neural information processing systems*, 36:80375–80395, 2023.

Andreas Buja, Werner Stuetzle, and Yi Shen. Loss functions for binary class probability estimation and classification: Structure and applications. *Technical report*, 2005.

Jonathan Chang, Masatoshi Uehara, Dhruv Sreenivas, Rahul Kidambi, and Wen Sun. Mitigating covariate shift in imitation learning via offline data with partial coverage. *Advances in Neural Information Processing Systems*, 34:965–979, 2021.

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

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

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

Alexander Irpan, Kanishka Rao, Konstantinos Bousmalis, Chris Harris, Julian Ibarz, and Sergey Levine. Off-policy evaluation via off-policy classification. *Advances in Neural Information Processing Systems*, 32, 2019.

Gunmin Lee, Wooseok Oh, Jeongwoo Oh, Seungyoun Shin, Dohyeong Kim, Jaeyeon Jeong, Sungjoon Choi, and Songhwai Oh. 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.

Nan Lu, Shida Lei, Gang Niu, Issei Sato, and Masashi Sugiyama. Binary classification from multiple unlabeled datasets via surrogate set classification. In *International Conference on Machine Learning*, pp. 7134–7144. PMLR, 2021.

Manu Orsini, Anton Raichuk, Léonard Hussenot, Damien Vincent, Robert Dadashi, Sertan Girgin, Matthieu Geist, Olivier Bachem, Olivier Pietquin, and Marcin Andrychowicz. What matters for adversarial imitation learning? *Advances in Neural Information Processing Systems*, 34:14656–14668, 2021.

Vaibhav Saxena, Matthew Bronars, Nadun Ranawaka Arachchige, Kuancheng Wang, Woo Chul Shin, Soroush Nasiriany, Ajay Mandlekar, and Danfei Xu. What matters in learning from large-scale datasets for robot manipulation. *arXiv preprint arXiv:2506.13536*, 2025.

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

Voot Tangkaratt, Nontawat Charoenphakdee, and Masashi Sugiyama. Robust imitation learning from noisy demonstrations. *arXiv preprint arXiv:2010.10181*, 2020.

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

Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

Yunke Wang, Chang Xu, Bo Du, and Honglak Lee. Learning to weight imperfect demonstrations. In *International Conference on Machine Learning*, pp. 10961–10970. PMLR, 2021.

Yunke Wang, Bo Du, and Chang Xu. Unlabeled imperfect demonstrations in adversarial imitation learning. *arXiv preprint arXiv:2302.06271*, 2023.

Yueh-Hua Wu, Nontawat Charoenphakdee, Han Bao, Voot Tangkaratt, and Masashi Sugiyama. Imitation learning from imperfect demonstration. In *International Conference on Machine Learning*, pp. 6818–6827. PMLR, 2019.

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

Chaopeng Zhang, Wenshuo Wang, Zhaokun Chen, and Junqiang Xi. 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.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

Brian D Ziebart, J Andrew Bagnell, and Anind K Dey. Modeling interaction via the principle of maximum causal entropy. In *International conference on machine learning*, pp. 1255–1262. Carnegie Mellon University, 2010.

Appendix

A Surrogate Demonstrator Classification training

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

Algorithm 3 Learning scoring model

1: **Input**: $\{(i, D_i)\}_{i=1}^m$, $\{\alpha_i\}_{i=1}^m$, and α'
2: **Initialization**: scoring model $f_\phi(s, a)$
3: Compute of $T(\cdot) = [T_1(\cdot), \cdots, T_m(\cdot)]$ using Eq. 5 based on $\{\alpha_i\}_{i=1}^m$ and α'
4: **for** $t = 1, 2, \cdots$ **do**
5: Sample examples: $\bar{z}, (s, a) \sim \{(i, D_i)\}_{i=1}^m$
6: Update f_ϕ by minimizing the empirical risk from Eq. 6
7: **end for**
8: **Return** f_ϕ

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), $f_{\phi^*_{\text{surr}}}$ that 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 minimize 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. 6.

B Proof of Theorem 1

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.

$$\mathcal{L}(\phi, \alpha) = \mathbb{E}_{(s,a)\sim\rho}\left[\ell(T_\alpha(f_\phi(s, a)), \bar{z})\right] \tag{9}$$

Minimizing the cross-entropy loss in Eq. 9 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

$$ll(\phi, \alpha) := \ln p(\bar{z}|(s, a), \phi) = \sum_\alpha p(\alpha)\ \ln p(\bar{z}|(s, a), \phi) \tag{10}$$

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 $p := p(\alpha|\bar{z}, (s, a), \phi)$

$$\begin{aligned}
L(q, \phi) &= \sum_\alpha q(\alpha) \ln \frac{p(\bar{z}, \alpha|(s, a), \phi)}{q(\alpha)} \\
&= \sum_\alpha q(\alpha) \left[\ln p(\alpha|\bar{z}, (s, a), \phi) + \ln p(\bar{z}|(s, a), \phi) - \ln q(\alpha)\right] \\
&= \sum_\alpha q(\alpha) \ln \frac{p(\alpha|\bar{z}, (s, a), \phi)}{q(\alpha)} + \sum_\alpha q(\alpha) \ln p(\bar{z}|(s, a), \phi) \\
&= -KL(q||p) + \ln p(\bar{z}|(s, a), \phi)
\end{aligned} \tag{11}$$

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

$$\begin{aligned} \ln p(d|(s,a),\phi) &= L(q,\phi) + KL(q||p), \\ &\geq L(q,\phi) \end{aligned} \tag{12}$$

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,

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

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.7. 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

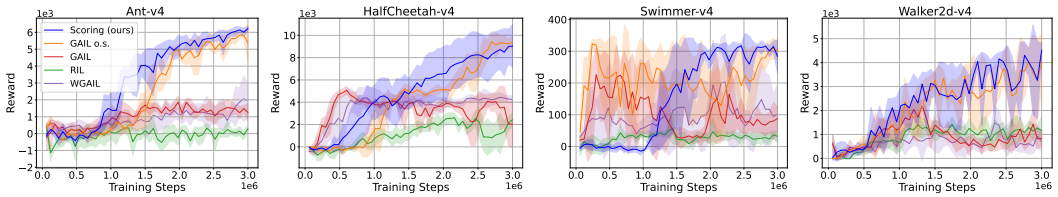
$$\begin{aligned} ll(\phi^t,\alpha^{t+1}) &= \ln p(\bar{z}|(s,a),\phi^t) \\ &= L(q(\alpha^{t+1}),\phi^t) + KL(q(\alpha^{t+1})||p(\alpha|\bar{z},(s,a),\phi^t)) \\ &= L(q(\alpha^{t+1}),\phi^t) \end{aligned} \tag{14}$$

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 $ll(\phi^{t+1},\alpha^{t+1}) \geq ll(\phi^t,\alpha^{t+1}) = \ln p(\bar{z}|(s,a),\phi^t) = ll(\phi^t,\alpha^t)$

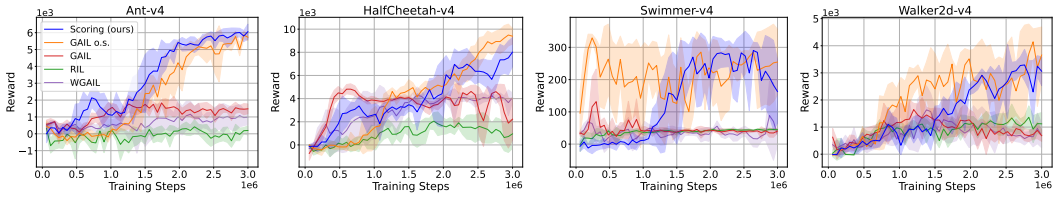
□

C Experiment Details

C.1 Detailed Results



(a) General expertise test



(b) Low expertise test

Figure 3: Performance comparison (mean ± 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 ± 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

Table 7: Estimated expertise levels $\{\hat{\alpha}_i\}_{i=1}^m$, showing the mean estimate and the mean ± 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 ± 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 ± 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 → 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 ± 3.9	14984.2 ± 10.1	15.8 ± 10.1	11.4 ± 3.9	0.9991 / 0.9989	-
	2	14984.0 ± 5.5	14984.0 ± 5.5	16.0 ± 5.5	16.0 ± 5.5	0.9989 / 0.9989	-0.02 / 0.00
HalfCheetah-v4	1	15000.0 ± 0.0	14999.4 ± 0.8	0.6 ± 0.8	0.0 ± 0.0	1.0000 / 1.0000	-
	2	15000.0 ± 0.0	15000.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	1.0000 / 1.0000	0.00 / 0.00
Swimmer-v4	1	15000.0 ± 0.0	14999.4 ± 1.2	0.6 ± 1.2	0.0 ± 0.0	1.0000 / 1.0000	-
	2	14991.4 ± 16.2	14991.4 ± 16.2	8.6 ± 16.2	8.6 ± 16.2	0.9994 / 0.9994	-0.06 / -0.06
Walker2d-v4	1	14992.0 ± 2.2	14985.0 ± 6.1	15.0 ± 6.1	8.0 ± 2.2	0.9992 / 0.9990	-
	2	14990.4 ± 1.6	14990.4 ± 1.6	9.6 ± 1.6	9.6 ± 1.6	0.9994 / 0.9994	0.02 / 0.04
Low expertise test							
Ant-v4	1	2997.6 ± 1.5	24101.4 ± 735.5	2898.6 ± 735.5	2.4 ± 1.5	0.9033 / 0.5178	-
	2	2947.0 ± 32.9	26947.0 ± 32.9	53.0 ± 32.9	53.0 ± 32.9	0.9965 / 0.9823	10.32 / 89.71
HalfCheetah-v4	1	3000.0 ± 0.0	25898.2 ± 355.8	1101.8 ± 355.8	0.0 ± 0.0	0.9633 / 0.7368	-
	2	2997.4 ± 5.2	26997.4 ± 5.2	2.6 ± 5.2	2.6 ± 5.2	0.9998 / 0.9991	3.79 / 35.60
Swimmer-v4	1	3000.0 ± 0.0	26156.2 ± 393.5	843.8 ± 393.5	0.0 ± 0.0	0.9719 / 0.7884	-
	2	2637.2 ± 290.4	26637.2 ± 290.4	362.8 ± 290.4	362.8 ± 290.4	0.9758 / 0.8791	0.40 / 11.50
Walker2d-v4	1	2965.0 ± 42.7	22953.2 ± 890.1	4046.8 ± 890.1	35.0 ± 42.7	0.8639 / 0.4301	-
	2	2772.0 ± 170.3	26772.0 ± 170.3	228.0 ± 170.3	228.0 ± 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 ± std of reward across last five checkpoints. Supplementary results for Sec. 4.5

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

Table 10: Ablation study on top-k hyperparameter on Ant-v4 task. Reported as mean ± std of reward across last five checkpoints. Supplementary results for Sec. 4.5

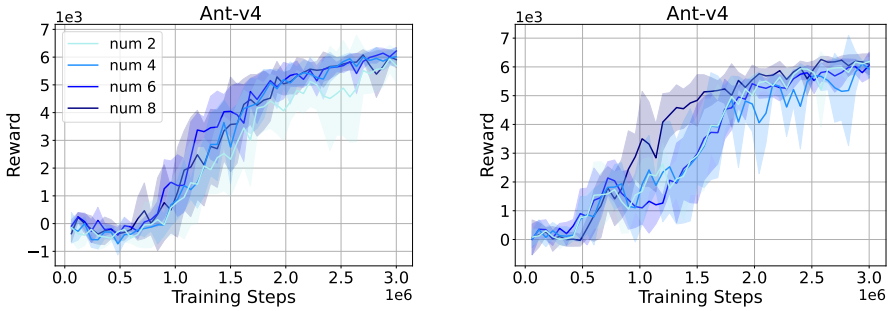
Top-k fraction	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
0.05	–	5887.6 ± 263.8
0.1	5403.8 ± 378.7	**5932.0 ± 335.2**
0.15	–	3663.4 ± 1950.9
0.2	–	1756.6 ± 902.6
0.3	5809.0 ± 274.8	–
0.5	**6084.6 ± 146.9**	–
0.55	4109.2 ± 2204.9	–
0.6	3641.0 ± 1237.7	–
0.7	2383.0 ± 1059.7	–

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

Demonstrator	Type	1	2	3	4	5	6	7	8	Mean error $\pm$ 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 for Ant-v4 task. Reported numbers are mean $\pm$ std of reward across last five checkpoints. Supplementary results for Sec. 4.5

Scoring Samples	Expertise [0.1 – 0.9]	Expertise [0.05 – 0.15]
2	5634.0 $\pm$ 617.8	6050.6 $\pm$ 120.4
4	5934.0 $\pm$ 264.3	5637.2 $\pm$ 772.5
6	**6084.6 $\pm$ 146.9**	5932.0 $\pm$ 335.2
8	5735.0 $\pm$ 225.6	**6146.6 $\pm$ 107.2**

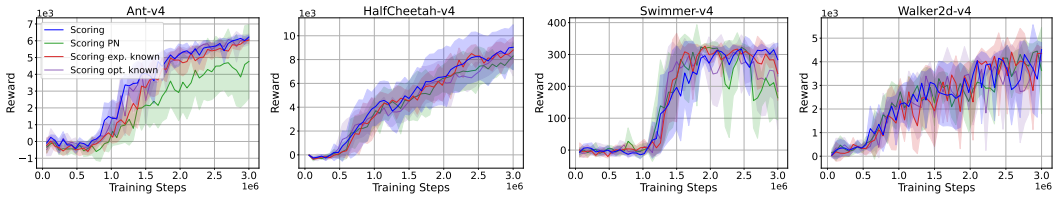


(a) Expertise Level range [0.1 - 0.9]

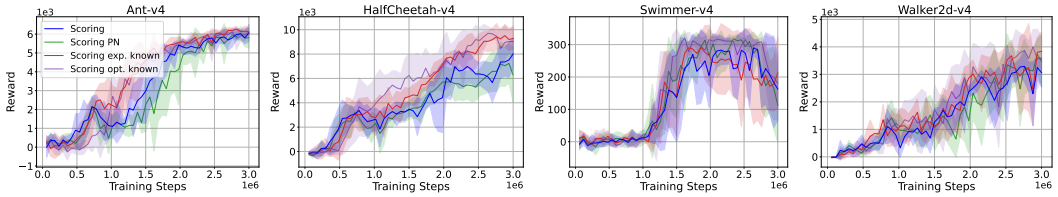
(b) Expertise Level range [0.05 - 0.15]

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

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.



(a) General expertise test



(b) Low expertise test

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

C.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. 8 is replaced with a PN loss $\widehat{\mathbb{E}}_{\substack{(s,a)\sim\hat{\rho}^{\pi^*}\\(s',a')\sim\hat{\rho}^{\pi^{\text{sub}}}}}\big[\mathcal{L}_{PN}(f_\phi(s,a,s',a')\big]$, 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 π^{sub} represent the pseudo-suboptimal data 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.

C.3 Reproducibility Statement

To ensure the reproducibility of this work, details of the training parameters are provided in Tab. 13. All reported results are averaged over five random seeds (0–4) per condition/method, with mean and standard deviation reported. The corresponding code will be released as open-source upon publication of this work.

Table 13: 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