Skill Disentanglement for Imitation Learning from Suboptimal Demonstrations

Tianxiang Zhao
the Pennsylvania State University
Pennsylvania, USA
tkz5084@psu.edu

Wenchao Yu
NEC Laboratories America
New Jersey, USA
wyu@nec-labs.com

Suhang Wang
the Pennsylvania State University
Pennsylvania, USA
szw494@psu.edu

Lu Wang
East China Normal University
Shanghai, China
luwang@stu.ecnu.edu.cn

Xiang Zhang
the Pennsylvania State University
Pennsylvania, USA
xzz89@psu.edu

Yuncong Chen
NEC Laboratories America
New Jersey, USA
yuncong@nec-labs.com

Yanchi Liu
NEC Laboratories America
New Jersey, USA
yanchi@nec-labs.com

Wei Cheng
NEC Laboratories America
New Jersey, USA
weicheng@nec-labs.com

Haifeng Chen
NEC Laboratories America
New Jersey, USA
haifeng@nec-labs.com

ABSTRACT

Imitation learning has achieved great success in many sequential decision-making tasks, in which a neural agent is learned by imitating collected human demonstrations. However, existing algorithms typically require a large number of high-quality demonstrations that are difficult and expensive to collect. Usually, a trade-off needs to be made between demonstration quality and quantity in practice. Targeting this problem, in this work we consider the imitation of sub-optimal demonstrations, with both a small clean demonstration set and a large noisy set. Some pioneering works have been proposed, but they suffer from many limitations, e.g., assuming a demonstration to be of the same optimality throughout time steps and failing to provide any interpretation w.r.t knowledge learned from the noisy set. Addressing these problems, we propose SDIL by evaluating and imitating at the sub-demonstration level, encoding action primitives of varying quality into different skills. Concretely, SDIL consists of a high-level controller to discover skills and a skill-conditioned module to capture action-taking policies, and is trained following a two-phase pipeline by first discovering skills with all demonstrations and then adapting the controller to only the clean set. A mutual-information-based regularization and a dynamic sub-demonstration optimality estimator are designed to promote disentanglement in the skill space. Extensive experiments are conducted over two gym environments and a real-world healthcare dataset to demonstrate the superiority of SDIL in learning from sub-optimal demonstrations and its improved interpretability by examining learned skills.

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08…$15.00
https://doi.org/10.1145/3580305.3599506

CCS CONCEPTS

• **Computing methodologies** → **Reinforcement learning**; **Learning from demonstrations**; *Semi-supervised learning settings.*

KEYWORDS

imitation learning; hierarchical reinforcement learning; skill discovery; noisy data

ACM Reference Format:
Tianxiang Zhao, Wenchao Yu, Suhang Wang, Lu Wang, Xiang Zhang, Yuncong Chen, Yanchi Liu, Wei Cheng, and Haifeng Chen. 2023. Skill Disentanglement for Imitation Learning from Suboptimal Demonstrations. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3580305.3599506

1 INTRODUCTION

Imitation learning (IL), which aims to imitate human demonstrations, has achieved great success in many sequential decision-making tasks, such as assistive robots [1, 2] and human-computer interactions [3, 4]. Generally, IL assumes access to a collected human demonstration set and learns the action policy by mimicking the latent generation process of those demonstrations [5–7]. Each demonstration is a sequence of transitions (state-action pairs), generated by experts in the target task. However, the success of most existing IL algorithms crucially depends on the availability of a large and high-quality demonstration set [8]; while in many real-world cases such as autonomous driving and healthcare treatments, it is challenging and expensive to collect them. Furthermore, humans often make mistakes due to various reasons such as difficulty of the task and partial observability [9]. The widely-adopted crowd-sourced data collection pipeline would also inevitably lead to demonstrations of varying levels of expertise [10]. The difficulty of obtaining large-scale high-quality demonstrations challenges many existing IL methods.

Therefore, in this work, we consider a more realistic setting, i.e., we have access to a small clean demonstration set in the accompany of a large noisy demonstration set. In the clean set, demonstrations

can be trusted to follow an optimal action-taking policy. On the contrary, the quality of a noisy demonstration can not be guaranteed and it may contain segments (consecutive transitions) of sub-optimal or even poor action selections. For example, in the healthcare domain (Fig. 1(a)), each demonstration contains the sequential medical treatment history of a particular patient with physiological features as states and medical treatments as actions. The clean set contains carefully examined records that are accurate and effective; while the noisy set is collected in the wild without expert scrutiny. Some noisy demonstrations could contain misdiagnosing, inappropriate use of medications, or dose miscalculations at certain stages. Directly incorporating the noisy set in learning may pollute the dataset and mislead the neural agent, resulting in a negative effect. However, optimal demonstrations are limited in quantity and could be insufficient for successful imitation learning. Hence, it is important to extract useful information from noisy demonstration to help train better IL models.

Extracting useful information from noisy demonstrations is non-trivial, as the quality of actions taken at each time step is not provided. There are some initial efforts on imitating sub-optimal demonstrations [9, 11, 12]. For example, CAIL [11] uses meta-learning to estimate the confidence over each noisy demonstration, DWBC [9] trains an additional discriminator to distinguish expert and non-expert demonstrations. However, all of these methods are coarse-grained, i.e., them conduct a trajectory-level evaluation and assign the same optimality score to actions across all time steps of the same demonstration [9, 11]. This is a strong assumption and limits their effectiveness in real-world applications. For instance, a maze-solving demonstration could contain both proper/efficient and redundant/inefficient segments simultaneously. Furthermore, previous methods lack interpretability. It is difficult to understand what they learn from clean and noisy demonstrations respectively, making the utilization of noisy demonstrations difficult to trust.

Addressing these challenges, we propose to learn a bi-level skill hierarchy from demonstrations to capture the variations of action policies. Each skill encodes a specific action-taking strategy in a part of the state-space [13], and each state-action pair of collected demonstrations can be deemed to be generated following a particular skill, as the example in Fig. 1(b). In modeling the distribution of demonstrations, high-quality segments of noisy demonstrations would have overlapped skill selections with clean demonstrations and can help improve the learning of those selected skills in turn. With this design, a set of skills of varying optimality can be extracted from modeling the distribution of demonstrations, and their optimality can be evaluated by analyzing skill-selection behaviors. Furthermore, each skill can be analyzed based on agent behaviors conditioned on it. By examining skill-selection distributions and comparing action-taking policies across skills, stronger interpretability w.r.t learned action primitives will be offered.

Concretely, we design a framework SDIL that comprises a high-level controller to encode observed states and make skill selections, and a low-level action predictor conditioned on the selected skill. SDIL is optimized in a two-phase pipeline. In the first phase, we discover latent skills from both clean and noisy demonstrations, with particularly designed regularization to promote skill disentanglement. With skills learned, in the second phase, we adapt

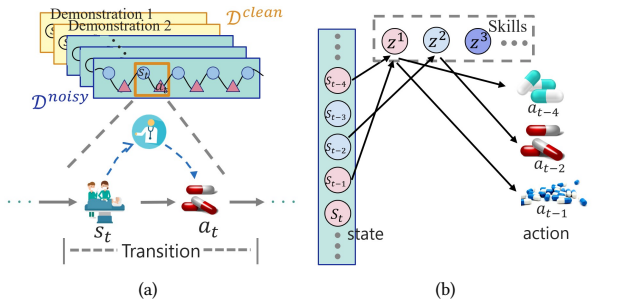


Figure 1: (a) An example in the healthcare domain. We have a small number of clean demonstrations and a large number of noisy demonstrations, each demonstration is composed of a sequence of transitions (state-action pairs). (b) High-level visualization of the skill hierarchy. Each skill encodes a mapping from states to actions. We can identify consecutive transitions modeled by the same skill as a segment with similar level of optimality.

this hierarchical framework by imitating only the clean demonstrations in order to compose an expert-level neural agent. Our main contributions are:

- We study a novel problem of imitation learning with sub-optimal demonstrations by extracting useful information from high optimality segments of noisy demonstrations.
- We propose a novel framework SDIL which can extract a set of skills from sub-optimal demonstrations and utilize them to compose an expert-level neural agent.
- Experiments are conducted on both synthetic and real-world datasets. Besides quantitative comparisons with existing algorithms, we also provide a set of case studies to analyze the behavior of SDIL and learned skills.

2 RELATED WORK

2.1 Imitation Learning

Imitation learning is a special case of reinforcement learning, which relies on expert trajectories composed of state-action pairs without availability of the interactive environment. Existing IL approaches can be folded into three paradigms, behavior cloning (BC), inverse reinforcement learning (IRL), and generative adversarial imitation learning (GAIL). BC [5] tackles this problem in a supervised manner, learning the action policy as a conditional distribution $\pi(\cdot \mid s)$ over actions. Works have observed that BC has no inferior performance compared to other popular IL algorithms such as GAIL [7] with expert demonstrations available [14, 15]. IRL [16] dedicates to recover the reward function which would be maximized by expert demonstrations, and learn the agent through exploration over the learned reward. Adversarial GAIL [7] takes an adversarial approach, train the policy model to generate trajectories that are indistinguishable from expert demonstrations. Adversarial inverse reinforcement learning (AIRL) [6] extends GAIL to simultaneously learn the reward function and the optimal policy, aiming to find the saddle point of a min-max optimization problem. However, these approaches take all demonstrations as equal and assume them to

be optimal. They cannot learn well in the existence of sub-optimal demonstrations.

Some existing works extend traditional IL algorithms to cope with imperfect demonstrations [17, 18]. However, most of these methods require prior domain knowledge, like rankings of demonstrations [19, 20] and demonstration confidence [21]. CAIL [11] utilizes a meta-learning framework to learn from sub-optimal demonstrations with a confidence estimator. In the inner-update step, it conducts a weighted imitation learning based on estimated confidence, and the estimator would be updated in the outer loop. DWBC [9] designs a discriminator to evaluate demonstration quality, and ILEED [12] proposes to estimate expertise of demonstrators. However, all these methods conduct a coarse-grained estimation, assuming all time steps of the same demonstration share the same optimality. Different from them, we propose to break each demonstration down and utilize those noisy demonstrations in a fine-grained manner.

2.2 Hierarchical Reinforcement Learning

Hierarchical reinforcement learning (HRL) decomposes the full control policy into multiple macro-operators or abstractions, each encoding a short-term decision process [22–25]. This hierarchical structure provides intuitive benefits for easier learning and long-term decision-making, as the policy is organized along the hierarchy of multiple levels of abstractions. Typically, the higher-level policy provides conditioning variables [13, 26, 27] or selected sub-goals (options) [28, 29] to control the behavior of lower-level policy models. Another branch of HRL methods design a hierarchy over the actions to reduce the search space with prior domain knowledge [23, 30].

A variety of discussions have been made over the trade-off between generality (wide applicability) and specificity (benefits for specific tasks) of low-level policies [28, 29, 31]. Recently, a particular advantage of introducing HRL has also been identified for improved exploration capabilities [32, 33]. In this work, we focus primarily on modeling the latent decision policies of sub-optimal demonstrations with a HRL-based framework, using a diverse skill set to model action primitives of varying optimality behind collected demonstrations. As far as we know, we are the first to consider hierarchical skill discovery for automatically learning from noisy demonstrations.

3 PROBLEM SETUP

In imitation learning, we aim to learn a policy π_θ from the collected demonstration set. Each demonstration τ is a trajectory, a sequence of transitions (state-action pairs): $\tau = (\boldsymbol{s}_0, \boldsymbol{a}_0, \boldsymbol{s}_1, \boldsymbol{a}_1, ...)$, with $\boldsymbol{s}_t \in \mathcal{S}$ and $\boldsymbol{a}_t \in \mathcal{A}$ being the state and action at time step t. $\mathcal{S}$ is the state space and $\mathcal{A}$ is the set of actions. A policy $\pi : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ maps the observed state to a probability distribution over actions. Most existing imitation learning works [9, 11] assume that these demonstrations are all optimal, collected from domain experts. However, it is usually challenging to collect a sufficiently large expert demonstration set. Hence, we often need to learn from noisy demonstrations, which challenge existing works.

Specifically, we have a small clean demonstration set $\mathcal{D}^{clean} = \{\tau_i\}_{i=1}^{n_e}$ which is drawn from the expert optimal policy π_e, and a noisy demonstration set $\mathcal{D}^{noisy} = \{\tau_i\}_{i=1}^{n_o}$ generated from some other behavioral policies. Note that the qualities of policies used by $\mathcal{D}^{noisy}$ are not evaluated, they could be of similar or much worse than the expert policy π_e. In this task, we want to learn a policy agent π_θ by extracting useful information from both optimal demonstrations $\mathcal{D}^{clean}$ and noisy demonstrations $\mathcal{D}^{noisy}$.

We assume that these demonstrations are generated from semantically meaningful skills, with each skill encoding a specific action primitive, a sub-policy. For example in the healthcare domain, each skill could represent a strategy of adopting treatment plans in the observance of several particular symptoms. Demonstrations in $\mathcal{D}^{noisy}$ can be split into multiple segments, and useful information can be extracted from segments that are generated from high-quality skills. Concretely, the task can be formalized as:

Given a small clean demonstration set $\mathcal{D}^{clean}$ and a large noisy demonstration $\mathcal{D}^{noisy}$, we aim to learn a policy agent π_θ for action prediction based on observed states: $\pi_\theta : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$

4 SKILL-BASED IL FROM IMPERFECT DEMONSTRATIONS

In this work, we propose to learn from sub-optimal demonstrations with a hierarchical framework, containing both a high-level policy for selecting skills and a low-level policy for action prediction. The high-level policy will maintain a skill set and select the to-be-used skills from observed states, and the low-level policy will decide on actions conditioned on the skill. This framework enables the automatic discovery of skills for utilizing sub-optimal demonstrations. To encourage the disentanglement of skills and obtain a well-performing agent π_θ, a two-phase framework is proposed: (1) the skill discovery phase utilizing $\mathcal{D}^{clean} \cup \mathcal{D}^{noisy}$ to extract and refine a skill set of varying optimality; (2) the skill reusing phase that adapts learned skills to imitate $\mathcal{D}^{clean}$, transferring the knowledge to learn expert policy π_θ. In this section, we will first present the hierarchical model architecture, and then introduce the learning of each phase one by one.

4.1 Hierarchical Policy for Skill Discovery

An overview of our hierarchical framework is shown in Figure 2, in which a set of variables $\{\boldsymbol{z}^k \in \mathbb{R}^{d_z}, k = [1, \ldots, K]\}$ is used to parameterize skills. d_z is the dimension of skill embeddings, and K is the total number of skill variables. The inference of SDIL follows two steps: (1) a high-level policy $\pi_{high}(\boldsymbol{z}_t \mid \ldots, \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{s}_t)$ that selects the skill $\boldsymbol{z}_t$ for time step t based on the historical transitions, corresponding to the skill encoding and skill matching modules in Fig. 2; (2) a low-level skill-conditioned policy module $\pi_{low}(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{z}_t)$ which predicts the to-be-taken actions, as in Fig. 2. Concretely, details of these three modules are introduced below:

- A skill encoding module that maps historical transitions and current states to the skill embedding space $\mathbb{R}^{d_z}$. We use $\boldsymbol{s}_t$ and a sequence of state-action pairs $[\boldsymbol{s}_{t-M}, \boldsymbol{a}_{t-M}, \ldots, \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}]$ as the input to obtain the latent skill embedding $\boldsymbol{z}'_t$. M is the length of the look-back window. To enable quick skill discovery in account of transition dynamics, we further incorporate states of the next step $\boldsymbol{s}_{t+1}$ as an auxiliary input during the skill discovery phase following [34], and the encoder can be modeled

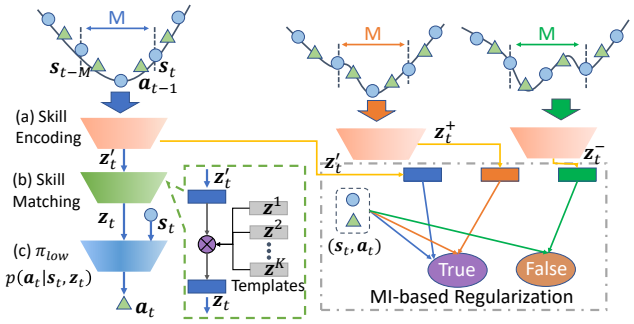


Figure 2: Our framework is composed of three modules: the skill encoding module, skill matching module g, and skill-conditioned policy module π_{low}. The skill encoding module uses previous time steps with window size M as input, and the skill matching module contains K skill templates as parameters. A mutual-information-based regularization is designed to promote automatic skill discovery. A novel skill-based positive-unlabeled (PU) learning along with deep embedding clustering (DEC) is designed to select positive and negative pairs (z^+ and z^-) for learning a skill set of varying optimality.

as $p_{f_{bi}}(z'_t \mid s_{t-M}, a_{t-M}, \ldots, s_t, s_{t+1})$. In the skill reusing phase, as future states should not available, we model the encoder as $p_{f_{uni}}(z'_t \mid s_{t-M}, a_{t-M}, \ldots, s_t)$.

- A skill matching module g that maintains a set of K prototypical embeddings $\{z^1, z^2, \ldots, z^K\}$ as K skills. In the inference of time step t, extracted skill embedding z'_t would be compared with these prototypes and get mapped to one of them to generate z_t, with the distribution probability following:

$$p(z_t = z^k) = \frac{1/D(z'_t, z^k)}{\sum_{i=1}^{K} 1/D(z'_t, z^i)}. \tag{1}$$

In this equation, $D(\cdot)$ is a distance measurement in the skill embedding space and we stick to the Euclidean distance in this work. To encourage the separation of skills and increase its interpretability, we use hard selection in the generation of z_t. However, gradients are not readily available for the selection operation. Addressing this problem, we adopt Gumbel softmax [35], in which the index of selected z is obtained following:

$$index_z = \arg\max_i \frac{\exp((G_i + \log(p(z_t = z^i)))/t)}{\sum_j \exp((G_j + \log(p(z_t = z^j)))/t)}, \quad G_i \sim Gumbel(0, 1). \tag{2}$$

G_i is sampled from the standard Gumbel distribution and t here represents temperature typically set to 1. The re-parameterization trick in [35] enables differentiable inference of Eq. 2, which allows these prototypical skill embeddings to be updated along with other parameters in the learning process.

- A low-level policy module π_{low} that captures the mapping from state to actions conditioned on the latent skill variable. It takes the state s_t (current observation) and skill variable z_t (skill to use) as the input, and predicts the action: $p_{\pi_{low}}(a_t \mid s_t, z_t)$. The imitation learning loss can be computed as:

$$\mathcal{L}_{imi} = -\mathbb{E}_{\tau_i}\mathbb{E}_{(s_t, a_t)\in\tau_i}\mathbb{E}_{z_t\sim\pi_{high}} \log p_{\pi_{low}}(a_t \mid s_t, z_t), \tag{3}$$

which takes a hierarchical structure and maximizes action prediction accuracy on given demonstrations.

The high-level policy π_{high} is modeled by bi-directional skill encoding module f_{bi} and skill matching module g in the first phase, and by uni-directional skill encoding module f_{uni} and g in the second phase. We will go into detail about learning of these two phases in the following sections.

4.2 Discovering the Skill Set

In the skill discovery phase, we target to imitate demonstrations of $\mathcal{D}^{clean} \cup \mathcal{D}^{noisy}$ with the designed hierarchical framework, modeling the dynamics in action-taking strategies with explicit skill variables. However, directly using the imitation loss in Eq. 3 alone is insufficient to learn a skill set of varying optimality. There are some further challenges:

- Each skill variable z^k may degrade to modeling an average of the global policy, instead of capturing distinct action-taking strategies from each other [36]. A sub-optimal high-level policy π_{high} could tend to select only a small subset of skills or query the same skill for very different states.
- As collected transitions are of varying qualities, the extracted skill set is expected to contain both high-quality skills and low-quality skills. However, ground-truth optimality scores of transitions from $\mathcal{D}^{noisy}$ are unavailable, rendering extra challenges in differentiating and evaluating these skills.

To address these challenges, we design a set of countermeasures. First, to encourage the discovery of specialized skills that are distinct from each other, we design a mutual-information-based regularization term. Second, to guide the skill selection and estimate segment optimality, we further refine the skill discovery process by deep clustering and skill optimality estimation with Positive-Unlabeled (PU) learning. In the end, we incorporate s_{t+1} during skill encoding to take the inverse skill dynamics into consideration. Next, we will go into their details in this section.

4.2.1 Mutual-Information-based Regularization. To encourage the discovery of distinct skills, we propose a mutual information (MI) based regularization in the skill discovery phase. Each skill variable z^k should encode a particular action policy, corresponding to the joint distribution of states and actions $p(s, a \mid z^k)$. From this observation, we propose to maximize the mutual information between the skill z and the state-action pair $\{s, a\}$: $\max I((s, a), z)$. Mutual information (MI) measures the mutual dependence between two variables from the information theory perspective, and formally it can be written as:

$$I((s, a), z) = \int_{\mathcal{S}\times\mathcal{A}}\int_{\mathcal{Z}} p(s, a, z) \cdot \log \frac{p(s, a, z)}{p(s, a) \cdot p(z)} d(s, a)dz, \tag{4}$$

in which $p(s, a, z)$ is the joint distribution probability, and $p(s, a)$ and $p(z)$ are the marginals. This mutual information objective can quantify how much can be known about (s, a) given z, or symmetrically, how much can be known about z given the transition (s, a). Maximizing this objective corresponds to encouraging each skill variable to encode an action-taking strategy that is identifiable, and maximizing the diversity of the learned skill set.

MI can not be readily computed for high-dimension data due to the probability estimation and integration in Eq. 4. Following [37–39], we design a JSD-based MI estimator and formulate the regularization term as follows:

$$\begin{aligned}\mathcal{L}_{mi} =& \mathbb{E}_{\tau_i}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\in\tau_i}[\mathbb{E}_{\boldsymbol{z}_i^+} sp(-T(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{z}_i^+)) \\ &+ \mathbb{E}_{\boldsymbol{z}_i^-} sp(T(\boldsymbol{s}_t,\boldsymbol{a}_t,\boldsymbol{z}_i^-))].\end{aligned} \tag{5}$$

In this equation, $T(\cdot)$ is a compatibility estimation function implemented as a MLP, and $sp(\cdot)$ is the *softplus* activation function. $\boldsymbol{z}_i^+$ represents the skill selected by $(\boldsymbol{s}_i, \boldsymbol{a}_i)$ that is a **positive pair** of $(\boldsymbol{s}_t, \boldsymbol{a}_t)$. On the contrary, $\boldsymbol{z}_i^-$ denotes the skill selected by $(\boldsymbol{s}_i, \boldsymbol{a}_i)$ that is a **negative pair** of $(\boldsymbol{s}_t, \boldsymbol{a}_t)$. A positive pair denotes the transition that is similar to $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ in both embedding and optimality quality, and contrarily is the negative pair. Details of obtaining these positive or negative pairs will be provided in Sec 4.2.2. This regularization can encourage different skill variables to encode different action policies, and those positive pairs should select similar skills while negative pairs should select different skills.

4.2.2 Positive and Negative Pair Discovery. The optimization of mutual information regularization in Eq. 5 requires obtaining positive and negative pairs to learn a diverse skill set. One direct solution would be using itself as the positive pair (use $\boldsymbol{z}_t$ in place of $\boldsymbol{z}_i^+$ in Eq. 5) and the skills used by randomly sampled other transitions as the negative pair, as conducted in [38]. However, such strategy neglects potential guiding information and may selects transitions using the same skill as negative pairs, introducing noises into the learning process. To promote the automatic skill discovery, instead of randomly sampling we propose to utilize two heuristics: similarity and estimated optimality of transitions.

Concretely, we design a dynamic approach for identifying positive and negative pairs based on those two heuristics. The proposed strategy is composed of two parts, (1) a deep clustering (DEC) algorithm that can discover latent groups of transitions and capture their similarities, which will encourage different skill variables to encode the action primitives of different transition groups; (2) a PU learning algorithm that utilizes both $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$ to evaluate the optimality of discovered skills, and can propagate estimated optimality scores to transitions. An illustration is provided in Fig. 3. Next, we will go into detail about these two algorithms.

To find similar transitions, we first propose to take a deep embedding cluster (**DEC**) strategy, by measuring the distance in the high-dimensional space extracted by the skill encoding module f_{bi}. Denoting the distance between $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ and $(\boldsymbol{s}_i, \boldsymbol{a}_i)$ as $D(\boldsymbol{z}_t', \boldsymbol{z}_i')$, we obtain the candidate positive group for $\boldsymbol{z}_t$ as the those transitions with a small distance from it, and candidate negative group as those transitions with a large distance. This design will encourage those transitions taken similarly by the skill encoding module to select similar skills, and contrarily for dissimilar ones. Note that measured distances in the embedding space could be very noisy at the beginning, and their quality would be gradually improved during training. Therefore, we add a proxy approach by applying the clustering algorithm directly to the input states, and use variable ζ to control the probability of adopting DEC or this pre-computed version. In training, we will gradually increase ζ to shift from this pre-computed clustering to DEC.

Next, we propose to obtain pseudo optimality scores and refine the candidate positive pairs with a Positive-Unlabeled (**PU**) learning scheme [40]. As $\mathcal{D}^{noisy}$ contains sub-optimal demonstrations with transitions taking imperfect actions, it is important to differentiate transitions of varying qualities and imitate them with different skills. However, one major challenge is that ground-truth evaluations of those transitions are unavailable. We only have transitions from $\mathcal{D}^{clean}$ as positive examples and transitions from $\mathcal{D}^{noisy}$ as unlabeled examples. Addressing it, we design a skill-based PU learning algorithm. Optimality scores of discovered skills will first be estimated and then be propagated to those unlabeled transitions.

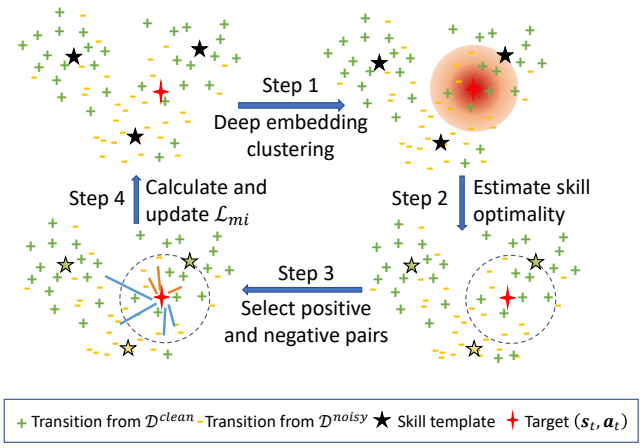


Figure 3: A high-level illustration of our strategy in selecting positive and negative pairs for $(\boldsymbol{s}_t, \boldsymbol{a}_t)$. In the first step, DEC is used to find transitions similar to the target (shown with a heatmap). Then, the optimality of skills will be estimated with PU learning (shown with different colors). Following that, estimated optimality scores would be propagated to transitions of $\mathcal{D}^{noisy}$, and our selection strategy is designed based on them (orange edges for positive pairs and blue edges for negative pairs).

Concretely, we first estimate the optimality score of skills based on the preference of expert demonstrations and the action prediction accuracy. Intuitively, those skills preferred by expert demonstrations over noisy demonstrations and have a high action prediction accuracy would be of higher qualityThen we propagate these scores to unlabeled transitions based on their skill selection distributions. The estimated optimality score will also evolve with the training process. The detailed algorithm steps are as follows:

- First, we obtain the skill selection distribution and denote it as $P^{\boldsymbol{z}} = \{p_k^{\boldsymbol{z}}, k \in [1, \ldots, K]\}$. We can calculate the selection distribution of clean demonstrations as:

$$p_k^{z,clean} = \mathbb{E}_{\tau_i\in\mathcal{D}^{clean}}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\in\tau_i} p(\boldsymbol{z}_t = \boldsymbol{z}^k), \tag{6}$$

  and of noisy demonstrations as:

$$p_k^{z,noisy} = \mathbb{E}_{\tau_i\in\mathcal{D}^{noisy}}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\in\tau_i} p(\boldsymbol{z}_t = \boldsymbol{z}^k). \tag{7}$$

- Then, the expert preference score s_k^{pref} of skill k can be computed as $(p_k^{z,clean} - p_k^{z,noisy})/(p_k^{z,clean} + \delta)$, in which δ is a small constant to prevent division by 0.

- The quality score of each skill can be computed based on its action-prediction accuracy when selected:

$$s_k^{qual} = \mathbb{E}_{\tau_i}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\in\tau_i}\mathbb{E}_{\boldsymbol{z}_t=\boldsymbol{z}^k} p(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{z}^k). \tag{8}$$

- The estimated optimality score s_k^{op} of skill k can be calculated by normalizing the product of these two scores $s_k^{pref} \cdot s_k^{qual}$ into the range $[-1, 1]$.
- With discovered skills evaluated, optimality scores will be propagated to each transition of $\mathcal{D}^{noisy}$, based on the skill it selected and its performance. For transition $(\boldsymbol{s}_t, \boldsymbol{a}_t)$, we compute its optimality as: $\sum_{k=1}^{K} p(z_t = z^k) \cdot s_k^{op}$.

All transitions of $\mathcal{D}^{clean}$ have optimality score set to 1. We would refine the candidate positive group of $\boldsymbol{z}_t$ by removing those that have a very different optimality score with a threshold ϵ. Note that this process is not needed for the candidate negative group, as they should be encouraged to select different skills regardless of optimality. The estimation of skill optimality scores is updated every N_{PU} epochs during training to reduce the instability. The algorithm is provided in Alg. 2.

4.2.3 Incorporating Next-step Dynamics. The core target of this work is to discover latent action-taking strategies from collected demonstrations and encode them explicitly. One problem is, as $\mathcal{D}^{noisy}$ is noisy and contains demonstrations of varying qualities, there may exist transitions taking very different actions with similar observations and histories. Due to the lack of ground-truth optimality score, it could be difficult for the skill encoding module to tell them apart and differentiate their latent skills. Therefore, in this skill discovery phase, we also include $\boldsymbol{s}_{t+1}$ as input to the skill encoding module, so that skills can be encoded in an influence-aware manner [34, 41]. $\boldsymbol{s}_{t+1}$ enables the skill selection to be not only conditioned on the current and prior trajectories, but also on a future state, which can help to differentiate skills that work in similar states. We denote this bi-directional skill encoder as f_{bi}, as in Fig. 2. Note that $\boldsymbol{s}_{t+1}$ is only used in the skill discovery phase, hence will not result in the information leakage problem.

Putting all together, in the skill discovery phase we will train modules $\{f_{bi}(\cdot), g(\cdot), \pi_{low}(\cdot)\}$ on $\mathcal{D}^{clean} \cup \mathcal{D}^{noisy}$, and use a mutual information loss to encourage the learning of a diverse skill set. Two algorithms are designed to estimate the similarity and optimality of transitions, and are used to refine the computation of the regularization term in Eq. 5. The full learning objective is:

$$\min_{f_{bi},g,\pi_{low}} \max_{T} \mathcal{L}_{imi} + \lambda \cdot \mathcal{L}_{mi}, \tag{9}$$

where T is the compatibility estimator used in MI estimation (Eq. 5). A detailed update step is provided in Alg. 2.

4.3 Reusing Skills for Expert Policy

With the skill discovery phase completed, we would utilize the learned skill set to imitate expert demonstrations in $\mathcal{D}^{clean}$. Specifically, we will adapt the modules $\{f_{uni}(\cdot), g(\cdot), \pi_{low}(\cdot)\}$ by imitating $\mathcal{D}^{clean}$ in this phase. Concretely, as $\{f_{bi}, g(\cdot), \pi_{low}(\cdot)\}$ are already learned in the skill discovery phase, we would split this *skill-reusing* into two steps: (1) In the first step, we would freeze the parameters of $\{g(\cdot), \pi_{low}(\cdot)\}$, which contain extracted skills and skill-conditioned policies, and only learn $\{f_{uni}(\cdot)\}$ on $\mathcal{D}^{clean}$ to obtain a high-level skill selection policy. This step utilizes pre-trained skills to mimic expert demonstrations $\mathcal{D}^{clean}$. Besides this imitation loss in Eq. 3, we further propose to transfer the skill selection knowledge from f_{bi} to f_{uni}:

$$\mathcal{L}_{KD} = -\mathbb{E}_{\tau_i}\mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\in\tau_i} p(\boldsymbol{z}_t = \bar{\boldsymbol{z}}_t), \tag{10}$$

in which $\bar{\boldsymbol{z}}_t$ is predicted using f_{bi}. For simplicity, we do not manipulate the weight of $\mathcal{L}_{KD}$ as it has the same scale as $\mathcal{L}_{imi}$, and the learning objective is

$$\min_{f_{uni},g,\pi_{low}} \mathcal{L}_{imi} + \mathcal{L}_{KD}. \tag{11}$$

(2) In the second step, we will further refine the whole framework in an end-to-end manner, based on the imitation objective $\mathcal{L}_{imi}$.

Aside from fine-tuning the skill-based framework on $\mathcal{D}^{clean}$, we further propose to utilize transitions from $\mathcal{D}^{noisy}$ that have a low optimality score [15]. In the skill discovery phase, a PU learning algorithm is conducted iteratively to evaluate the quality of transitions from $\mathcal{D}^{noisy}$, and will assign an optimality score to each of them. We can collect transitions with low optimality scores from demonstrations in $\mathcal{D}^{noisy}$ as $\mathcal{D}^{neg}$, and a new optimization objective $\mathcal{L}_{adv}$ can be designed to encourage our agent to perform differently:

$$\min_{f_{uni},g,\pi_{low}} \mathcal{L}_{adv} = \mathbb{E}_{(\boldsymbol{s}_t,\boldsymbol{a}_t)\in\mathcal{D}^{neg}} \log p(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{z}_t). \tag{12}$$

In experiments, we set a hard threshold to collect $\mathcal{D}^{neg}$, and the learning objective becomes $\mathcal{L}_{imi} + \mathcal{L}_{adv}$. This objective can encourage the model to prevent making the same mistakes as those low-quality demonstrations.

Algorithm 1 Skill-based IL from Imperfect Demonstrations

Input: Expert demonstrations $\mathcal{D}^{clean}$, Noisy demonstrations $\mathcal{D}^{noisy}$, Pre-train epochs N, PU interval N_{PU}

1: Random initialize model parameters
2: **for** t in N **do**
3: Optimize f_{bi}, g and π_{low} to discover skills following Alg. 2
4: **if** $t\%N_{PU} = 0$ **then**
5: Update optimality estimation of skills, as Sec. 4.2.2
6: **end if**
7: **end for**
8: **while** Not Converged **do**
9: Freeze parameters in $g(\cdot)$ and $\pi_{low}(\cdot)$
10: Learn f_{uni} with $\mathcal{L}_{imi} + \mathcal{L}_{KD}$ on $\mathcal{D}^{clean}$
11: **end while**
12: **while** Not Converged **do**
13: Update f_{uni}, g and π_{low} with $\mathcal{L}_{imi} + \mathcal{L}_{adv}$
14: **end while**
15: **return** Trained f_{uni}, g and π_{low}.

4.4 Training Algorithm

The training algorithm is provided in Alg. 1 and Alg. 2. The skill discovery phase corresponds to line 2 – 7 in Alg. 1, and the details of learning with MI-based regularization in line 3 is presented in Alg. 2. Mi-based regularization will help SDIL to learn a set of disentangled skills. Then, in the skill reusing phase, we first freeze the learned

Algorithm 2 MI-augmented Skill Discovery Step

Input: Transition set $\{(\boldsymbol{s}_t, \boldsymbol{a}_t)\}$ from $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$, skill encoding module f_{bi}, skill matching module g, skill-conditioned policy model π_{low}, MI compatibility estimation function T

1: Draw b transition samples $\{(\boldsymbol{s}_t, \boldsymbol{a}_t)\}_{t=1}^{b} \sim \mathcal{D}^{clean} \cup \mathcal{D}^{noisy}$
2: For each $(\boldsymbol{s}_t, \boldsymbol{a}_t)$, sample its candidate positive pairs $(\boldsymbol{s}_t^+, \boldsymbol{a}_t^+)$ from the same clustering group
3: Filter candidate positive pairs based on estimated optimality score following Sec. 4.2.2
4: For each $(\boldsymbol{s}_t, \boldsymbol{a}_t)$, sample its negative pairs $(\boldsymbol{s}_t^-, \boldsymbol{a}_t^-)$ of different clustering groups
5: Estimate the mutual information loss $\mathcal{L}_{mi}$ following Eq. 5
6: Update compatibility function T as $T \leftarrow T + \nabla_T \mathcal{L}_{mi}$
7: Update f_{bi}, g and π_{low} by: $\min_{f_{bi},g,\pi_{low}} \mathcal{L}_{imi} + \lambda \cdot \mathcal{L}_{mi}$
8: **return** Updated f_{bi}, g and π_{low}, T

skills to update the high-level policy in line 8 − 11, and then we finetune the whole framework end-to-end in line 12 − 14.

5 EXPERIMENT

In this section, we conduct experiments to evaluate the proposed imitation learning framework in discovering skills and learning high-quality policies from sub-optimal demonstrations. Specifically, in these experiments, we want to answer the following questions:

- **RQ1**: Can our framework SDIL achieve stronger performance in imitating demonstrations of varying qualities?
- **RQ2**: Is the proposed skill discovery algorithm able to disentangle latent skills of varying optimality apart?
- **RQ3**: Can SDIL provide improved interpretability on the knowledge learned from demonstrations?

5.1 Experiment Settings

5.1.1 Datasets. We evaluate the proposed framework on three datasets, MiniGrid-FourRoom, DoorKey [42], along with a public EHRs dataset Mimic-IV [43]. In FourRoom and Doorkey, the agent needs to navigate in a maze and arrive at the target position to receive a reward. In Mimic-IV, each demonstration is a trajectory of medication actions received by a patient, with physiological features as states and treatments as actions. Details of these datasets are provided in Appendix. A.

5.1.2 Baselines. First, we compare SDIL to representative imitation learning and treatment recommendation methods:

- Behavior Cloning (BC) [5]: BC bins the demonstrations into transitions and learns an action-taking policy in a supervised manner.
- DensityIRL [44]: Density-based reward modeling conducts a non-parametric estimation of marginal as well as joint distribution of states and actions from provided demonstrations, and computes a reward using the log of those probabilities.
- MCE IRL [45]: Its full name is Maximum causal entropy inverse reinforcement learning, whose idea is similar to DensityIRL. It designs an entropy-based approach to estimate causally conditioned probabilities for the estimation of reward function.
- AIRL [6]: Adversarial inverse reinforcement learning aims to automatically acquiring a scalable reward function from expert demonstrations by adversarial learning.
- GAIL [7]: GAIL directly trains a discriminator adversarially to provide the reward signal for policy learning.

The above methods are all proposed assuming demonstrations to be clean and optimal. We further compare SDIL with two recent strategies that are designed to utilize noisy demonstrations explicitly: ACIL [15] and DWBC [9].

- ACIL [15]: ACIL extends GAIL by incorporating noisy demonstrations as negative examples and learns the policy model to be distinct from them.
- DWBC [9]: It trains a discriminator to evaluate the quality of each demonstration trajectory through adversarial training, and conducts a weighted behavior cloning.

Different from them, our method proposes to break each trajectory down into segments and extract useful skills from them hierarchically. For the ablation study, we also implement a variant:

- SDIL*: This variant does not access $\boldsymbol{s}_{t+1}$ in the skill discovery, to analyze the importance of incorporating next-step dynamics.

5.1.3 Evaluation Metrics. We evaluate the trained agent with two strategies, the data-dependent and environment-dependent respectively. Concretely, the adopted metrics are as follows: (1) data-dependent metrics, which are computed on the test set of clean demonstrations. We select accuracy (ACC), macro AUROC, and micro AUROC scores [46]. For the FourRoom dataset, some actions are never taken so we report the Macro F score instead. (2) Environment-dependent metrics. For datasets with the interactive environment readily available, we place the trained agent into the environment and report its averaged rewards after 1,000 rounds of random running.

5.1.4 Configurations. The policy agent is instantiated to have the same network architecture across different methods for a fair comparison. All methods are trained until convergence, with the learning rate initialized to 0.001. Adam [47] optimizer is applied for all approaches with weight decay set to $5e$-4. Of obtained demonstrations, train:validation:test split is set as 3 : 2 : 5. For the proposed SDIL, the hyper-parameter λ which controls the weight of $\mathcal{L}_{mi}$ is set to 1.0 and the threshold ϵ of pseudo optimality score from PU learning is set to 0.1 if not stated otherwise. The look-back window size M is fixed as 5, and the controlling variable ζ of DEC is initialized to 0 and increased by 0.05 after each training epoch.

5.2 Imitation Learning Performance

First, we report the performance for action prediction and make the comparison in Table 1. to answer RQ1. Each experiment is run for 5 times with random initializations, and we report both the mean and variance w.r.t each metric. Two settings are adopted:

- $\mathcal{D}^{clean}$ Only: in the first setting, we test all approaches only on the expert demonstration set, $\mathcal{D}^{clean}$, without utilizing those noisy demonstrations. In this setting, all demonstrations can be trusted to be well-performing.
- $\mathcal{D}^{clean} \bigcup \mathcal{D}^{noisy}$: In the second setting, we use the mixed dataset containing both $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$ to augment the training

Table 1: Results for FourRoom, Mimic-IV and DoorKey. Best performance is highlighted in bold.

	Method	FourRoom			Mimic-IV				DoorKey		
		ACC	MacroF	Reward	ACC	MacroAUC	MicroAUC	MacroF	ACC	MacroF	Reward
$\mathcal{D}^{\text{clean}}$ **Only**	BC	$89.7_{\pm 0.26}$	$67.2_{\pm 0.18}$	$27.6_{\pm 0.9}$	$43.7_{\pm 0.41}$	$80.7_{\pm 0.22}$	$83.3_{\pm 0.18}$	$16.7_{\pm 0.15}$	$88.4_{\pm 0.34}$	$85.8_{\pm 0.21}$	$83.4_{\pm 2.1}$
	DensityIRL	$86.4_{\pm 0.27}$	$66.5_{\pm 0.22}$	$18.2_{\pm 1.5}$	$38.5_{\pm 0.28}$	$77.9_{\pm 0.16}$	$80.8_{\pm 0.19}$	$14.2_{\pm 0.14}$	$81.7_{\pm 0.22}$	$83.7_{\pm 0.16}$	$75.5_{\pm 3.4}$
	MCE IRL	$88.9_{\pm 0.26}$	$67.3_{\pm 0.19}$	$19.8_{\pm 1.6}$	$41.1_{\pm 0.21}$	$79.4_{\pm 0.16}$	$81.9_{\pm 0.17}$	$15.9_{\pm 0.17}$	$84.5_{\pm 0.29}$	$86.2_{\pm 0.23}$	$78.9_{\pm 2.5}$
	AIRL	$87.1_{\pm 0.22}$	$65.8_{\pm 0.23}$	$23.8_{\pm 1.2}$	$41.5_{\pm 0.36}$	$80.2_{\pm 0.23}$	$82.8_{\pm 0.18}$	$15.8_{\pm 0.16}$	$89.2_{\pm 0.26}$	$86.5_{\pm 0.25}$	$85.7_{\pm 2.7}$
	GAIL	$84.5_{\pm 0.33}$	$63.2_{\pm 0.31}$	$18.9_{\pm 1.1}$	$39.2_{\pm 0.31}$	$76.5_{\pm 0.21}$	$81.3_{\pm 0.19}$	$14.6_{\pm 0.18}$	$80.6_{\pm 0.29}$	$81.3_{\pm 0.27}$	$73.3_{\pm 3.2}$
	SDIL	$91.2_{\pm 0.21}$	$68.1_{\pm 0.16}$	$28.2_{\pm 1.2}$	$44.6_{\pm 0.32}$	$81.3_{\pm 0.26}$	$83.6_{\pm 0.21}$	$17.2_{\pm 0.18}$	$89.6_{\pm 0.24}$	$87.3_{\pm 0.22}$	$86.1_{\pm 2.3}$
$\mathcal{D}_{clean} \cup \mathcal{D}_{noisy}$	BC	$87.2_{\pm 0.24}$	$66.4_{\pm 0.15}$	$26.7_{\pm 1.0}$	$41.1_{\pm 0.36}$	$81.2_{\pm 0.19}$	$82.9_{\pm 0.16}$	$16.5_{\pm 0.16}$	$86.7_{\pm 0.24}$	$83.6_{\pm 0.19}$	$81.8_{\pm 2.3}$
	DensityIRL	$85.4_{\pm 0.27}$	$65.4_{\pm 0.26}$	$16.5_{\pm 1.3}$	$36.7_{\pm 0.29}$	$77.6_{\pm 0.17}$	$79.4_{\pm 0.21}$	$13.7_{\pm 0.15}$	$76.8_{\pm 0.25}$	$75.1_{\pm 0.19}$	$71.6_{\pm 2.6}$
	MCE IRL	$87.4_{\pm 0.25}$	$66.3_{\pm 0.21}$	$18.7_{\pm 1.5}$	$39.5_{\pm 0.27}$	$78.9_{\pm 0.18}$	$81.2_{\pm 0.19}$	$14.5_{\pm 0.14}$	$81.3_{\pm 0.23}$	$81.4_{\pm 0.23}$	$74.8_{\pm 2.2}$
	AIRL	$88.7_{\pm 0.28}$	$66.7_{\pm 0.33}$	$25.4_{\pm 1.2}$	$40.7_{\pm 0.33}$	$80.4_{\pm 0.23}$	$82.1_{\pm 0.17}$	$15.7_{\pm 0.16}$	$87.5_{\pm 0.31}$	$84.9_{\pm 0.28}$	$82.1_{\pm 2.2}$
	GAIL	$83.4_{\pm 0.29}$	$62.9_{\pm 0.26}$	$18.8_{\pm 1.1}$	$38.1_{\pm 0.29}$	$74.8_{\pm 0.22}$	$80.1_{\pm 0.15}$	$13.9_{\pm 0.17}$	$71.7_{\pm 0.22}$	$70.8_{\pm 0.17}$	$70.3_{\pm 2.4}$
	ACIL	$90.2_{\pm 0.23}$	$67.6_{\pm 0.19}$	$27.5_{\pm 1.1}$	$44.2_{\pm 0.35}$	$81.2_{\pm 0.24}$	$83.7_{\pm 0.18}$	$17.1_{\pm 0.15}$	$89.2_{\pm 0.30}$	$87.3_{\pm 0.22}$	$85.9_{\pm 2.8}$
	DWBC	$91.1_{\pm 0.17}$	$67.9_{\pm 0.22}$	$28.3_{\pm 2.1}$	$45.1_{\pm 0.37}$	$81.4_{\pm 0.25}$	$84.3_{\pm 0.17}$	$17.2_{\pm 0.16}$	$90.3_{\pm 0.27}$	$88.5_{\pm 0.26}$	$87.1_{\pm 3.3}$
	SDIL*	$93.4_{\pm 0.18}$	$69.1_{\pm 0.12}$	$31.2_{\pm 1.1}$	$48.1_{\pm 0.31}$	$83.8_{\pm 0.21}$	$86.5_{\pm 0.13}$	$20.7_{\pm 0.16}$	$92.9_{\pm 0.16}$	$91.4_{\pm 0.25}$	$92.3_{\pm 2.0}$
	SDIL	$\mathbf{94.4_{\pm 0.16}}$	$\mathbf{69.5_{\pm 0.11}}$	$\mathbf{33.5_{\pm 1.2}}$	$\mathbf{49.4_{\pm 0.26}}$	$\mathbf{84.4_{\pm 0.17}}$	$\mathbf{87.6_{\pm 0.12}}$	$\mathbf{22.3_{\pm 0.14}}$	$\mathbf{94.6_{\pm 0.21}}$	$\mathbf{92.6_{\pm 0.18}}$	$\mathbf{93.2_{\pm 2.2}}$

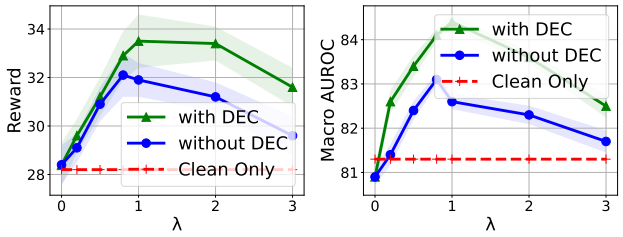
set. This setting can verify whether an algorithm can effectively leverage non-expert demonstrations.

It can be observed that our proposed SDIL outperforms all baseline algorithms, especially in the second setting, showing its strong performance in effectively utilizing the noisy demonstrations in addition to the expert ones. For both datasets, incorporating $\mathcal{D}^{noisy}$ would impair the performance of most baseline algorithms with a clear performance gap that can be observed. Such phenomenon suggests that directly imitating $\mathcal{D}^{noisy}$ is undesired and may be misled by noisy demonstrative trajectories. ACIL neglects useful information from $\mathcal{D}^{noisy}$ and takes all of them as negative, hence only achieving marginal improvements. DWBC adopts adversarial training to evaluate trajectory quality and identifies high-quality demonstrations, however it takes each trajectory as a whole and shows only marginal improvement. In the contrast, our method SDIL is able to make better use of $\mathcal{D}^{noisy}$, and shows stronger improvements across various metrics on both datasets. Furthermore, the comparison with variant SDIL* validates the effectiveness of incorporating next-step dynamics as auxiliary evidence for the discovery of skills, showing constant improvements across all datasets.

5.3 Ablation Study

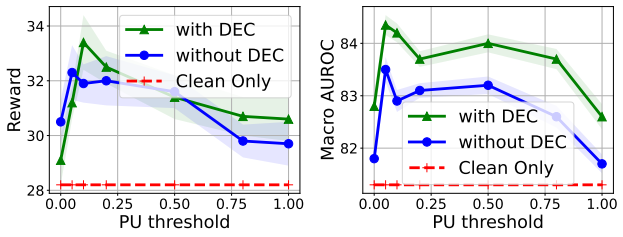
Skill Discovery With Mutual Information. In SDIL, an MI-based regularization is adopted to encourage the discovery of concrete skills. To evaluate the importance of designed mutual information regularization and partially answer RQ2, we conduct a sensitivity analysis on λ, which controls the weight of $\mathcal{L}_{mi}$ in the skill discovery phase. Concretely, we vary λ in $\{0, 0.2, 0.5, 0.8, 1, 2, 3\}$ and leave other hyper-parameters unchanged. Results on both the FourRoom dataset and Mimic-IV are reported in Fig. 4. As a comparison, we show the result w/o DEC part, and also present the performance of the base method trained only with $\mathcal{D}^{clean}$.

From Fig. 4, it can be observed that when λ is 0, in which case mutual information-based regularization is not applied in the skill discovery phase, the framework is ineffective to extract useful knowledge from noisy demonstrations. Generally, increasing λ



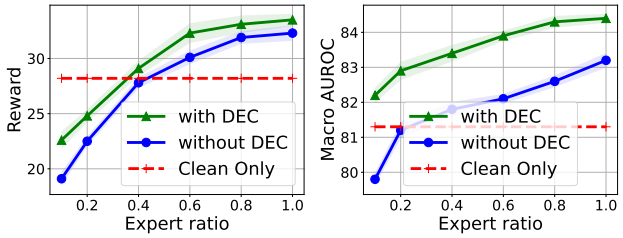
(a) FourRoom, Reward (b) Mimic-IV, macroAUC

Figure 4: Sensitivity analysis on mutual information regularization $\mathcal{L}_{mi}$ by varying its weight λ.



(a) FourRoom, Reward (b) Mimic-IV, macroAUC

Figure 5: Analysis on PU learning by varying the threshold ϵ for pseudo optimality estimation.



(a) FourRoom, Reward (b) Mimic-IV, macroAUC

Figure 6: Analysis on sizes of $\mathcal{D}^{clean}$ by taking different ratios of training examples.

would improve model performance within the range $[0, 1]$. When λ becomes larger than 2, a negative correlation with the performance can be observed. This phenomenon may arise that mutual information loss $\mathcal{L}_{mi}$ dominates the skill discovery process, resulting in learned skills performing poorly w.r.t imitation learning loss $\mathcal{L}_{imi}$. Generally, setting λ within $[0.8, 1.5]$ shows strong performance for both datasets and outperforms the baseline with only clean demonstrations with a clear margin, and applying DEC can further achieve constant improvements.

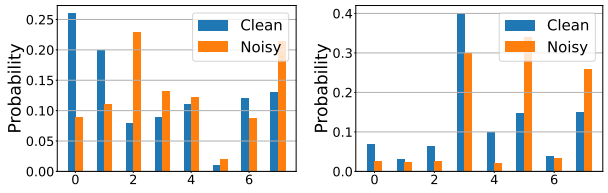
Utilizing $\mathcal{D}^{noisy}$ With PU Learning. To utilize both $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$ by discovering skills of varying qualities, we incorporate PU learning to augment the skill discovery together with the DEC mechanism. In particular, a hyper-parameter ϵ is introduced as the threshold of estimated optimality scores, as shown in Sec. 4.2.2. In this part to answer RQ2, we evaluate model performance with threshold ϵ varying in $\{0, 0.1, 0.2, 0.4, 0.5, \}$. A larger ϵ indicates fewer transitions would be identified as trustworthy to be positive/negative, while a smaller ϵ would result in utilizing more of obtained optimality scores from PU learning but may also submit to more noises meanwhile.

Experiments are conducted with other configurations unchanged and results are visualized in Fig. 5. Again, we implement two variants w/o the DEC part, and present the base method trained only on $\mathcal{D}^{clean}$ for a comparison. It is shown that for dataset FourRoom, ϵ is recommended to be set around 0.1. For Mimic-IV, SDIL performs relatively stable with ϵ within the range $[0.1, 0.5]$. Setting the threshold too low would introduce more noise while setting it too high would result in information loss. Furthermore, setting ϵ too low or too high would also reduce the significance of introducing Deep Embedding Clustering in positive/negative pair selection.

Influence of $\mathcal{D}^{clean}$ size The benefits of introducing $\mathcal{D}^{noisy}$ to augment the training data would be largely affected by the number of expert demonstrations available. To examine the improvement of SDIL in such scenarios, in this part we only take a subset of training demonstrations from $\mathcal{D}^{clean}$ as available. For the training set of expert demonstrations, we further vary its availability ratio in $\{0.1, 0.2, 0.4, 0.6, 0.8, 1\}$. All other hyper-parameters remain unchanged, and we report the results in Fig. 6. It can be observed that the model performance would increase with the expert demonstration ratio, and the DEC part shows a constant improvement. Furthermore, it can be observed that our proposed SDIL can outperform the baseline (using only the clean demonstration set $\mathcal{D}^{clean}$) with half the number of clean demonstrations.

Tendency of Skill Selection Behaviors. In this part, we want to evaluate the ability of SDIL to identify skills of varying quality by comparing the skill selection behaviors of demonstrations in $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$ statistically. Concretely, we test the trained model on each demonstration and compute the probability of selecting each skill per time step. Skill selection distributions are averaged per demonstration set at each time step in an unweighted manner, and a comparison is made for both the FourRoom dataset (Fig. 7(a)) and the Mimic-IV dataset (Fig. 7(b)).

In Fig. 7(a) and Fig. 7(b), a clear distribution gap can be observed in the skill selection behavior during imitating demonstrations in $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$. This result verifies the ability of SDIL to extract a skill set of varying optimality, that some are preferred



(a) FourRoom (b) Mimic-IV

Figure 7: Comparison of skill selection preference between expert demonstration set and noisy demonstration set on dataset FourRoom. x-axis represents the skill variables and y-axis is the selection probability.

in imitating expert demonstrations while some are preferred by noisy demonstrations. This design also enables effective knowledge sharing across demonstration sets, like the skill 3 in Fig. 7(b) which are selected frequently by both $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$.

5.4 Case Study

To provide deeper analysis of the proposed skill discovery process and answer RQ3, we provide a set of case studies on FourRoom and Mimic-IV (Appendix. C) to interpret skills learned by SDIL. Due to the limitation of space, we put results in Appendix. C.

6 CONCLUSION AND FUTURE WORK

In this paper, we propose an effective framework SDIL for imitation learning from sub-optimal demonstrations, which is able to utilize both clean and noisy demonstrations simultaneously. Taking a hierarchical architecture, SDIL manages to learn a disentangled skill set of varying optimality and compose an expert neural agent with them. SDIL also offers improved interpretability by analyzing learned skills. Breaking demonstrations down, SDIL can extract knowledge and utilize noisy demonstrations at the segment level, instead of assuming all time steps of the same demonstration to be of the same quality. Experimental results validate the effectiveness of SDIL, and comprehensive analysis is provided for ablation study and examination of learned skills. Note that the core of our method lies upon the discovered skill set.

As part of future work, we plan to explore the transferability of discovered skills, considering the discovery and manipulation of sub-policy skills with multiple different objectives.

7 ACKNOWLEDGMENTS

This material is based upon work partially supported by National Science Foundation (NSF) under grant number IIS-1909702 and the Army Research Office (ARO) under grant number W911NF21-1-0198 to Suhang Wang.

REFERENCES

[1] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[2] J. Hemminghaus and S. Kopp, "Towards adaptive social behavior generation for assistive robots using reinforcement learning," in *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*, 2017, pp. 332–340.

[3] T. Zhao, L. Liu, G. Huang, H. Li, Y. Liu, L. GuiQuan, and S. Shi, "Balancing quality and human involvement: An effective approach to interactive neural machine translation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 9660–9667.

[4] G. Huang, L. Liu, X. Wang, L. Wang, H. Li, Z. Tu, C. Huang, and S. Shi, "Transmart: A practical interactive machine translation system," *arXiv preprint arXiv:2105.13072*, 2021.

[5] D. Pomerleau, "An autonomous land vehicle in a neural network," *Advances in Neural Information Processing Systems*, vol. 1, 1998.

[6] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adverserial inverse reinforcement learning," in *International Conference on Learning Representations*, 2018.

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

[8] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 3406–3413.

[9] H. Xu, X. Zhan, H. Yin, and H. Qin, "Discriminator-weighted offline imitation learning from suboptimal demonstrations," in *International Conference on Machine Learning*. PMLR, 2022, pp. 24 725–24 742.

[10] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 1048–1055.

[11] S. Zhang, Z. Cao, D. Sadigh, and Y. Sui, "Confidence-aware imitation learning from demonstrations with varying optimality," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[12] M. Beliaev, A. Shih, S. Ermon, D. Sadigh, and R. Pedarsani, "Imitation learning by estimating expertise of demonstrators," *arXiv preprint arXiv:2202.01288*, 2022.

[13] A. Sharma, S. Gu, S. Levine, V. Kumar, and K. Hausman, "Dynamics-aware unsupervised discovery of skills," in *International Conference on Learning Representations*, 2019.

[14] Y. J. Ma, "From adversarial imitation learning to robust batch imitation learning," Ph.D. dissertation, 2020.

[15] L. Wang, W. Yu, X. He, W. Cheng, M. R. Ren, W. Wang, B. Zong, H. Chen, and H. Zha, "Adversarial cooperative imitation learning for dynamic treatment regimes," in *Proceedings of The Web Conference 2020*, 2020, pp. 1785–1795.

[16] B. D. Ziebart, A. L. Maas, J. A. Bagnell, A. K. Dey *et al.*, "Maximum entropy inverse reinforcement learning." in *Aaai*, vol. 8. Chicago, IL, USA, 2008, pp. 1433–1438.

[17] M. Yang, S. Levine, and O. Nachum, "Trail: Near-optimal imitation learning with suboptimal data," in *International Conference on Learning Representations*, 2021.

[18] F. Sasaki and R. Yamashina, "Behavioral cloning from noisy demonstrations," in *International Conference on Learning Representations*, 2020.

[19] H. Sugiyama, T. Meguro, and Y. Minami, "Preference-learning based inverse reinforcement learning for dialog control," in *Thirteenth Annual Conference of the International Speech Communication Association*, 2012.

[20] C. Wirth, J. Fürnkranz, and G. Neumann, "Model-free preference-based reinforcement learning," in *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

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

[22] P. Dayan and G. E. Hinton, "Feudal reinforcement learning," *Advances in neural information processing systems*, vol. 5, 1992.

[23] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[24] A. S. Vezhnevets, S. Osindero, T. Schaul, N. Heess, M. Jaderberg, D. Silver, and K. Kavukcuoglu, "Feudal networks for hierarchical reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2017, pp. 3540–3549.

[25] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 31, no. 1, 2017.

[26] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," *arXiv preprint arXiv:1802.06070*, 2018.

[27] V. Campos, A. Trott, C. Xiong, R. Socher, X. Giró-i Nieto, and J. Torres, "Explore, discover and learn: Unsupervised discovery of state-covering skills," in *International Conference on Machine Learning*. PMLR, 2020, pp. 1317–1327.

[28] M. Klissarov and D. Precup, "Flexible option learning," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[29] V. Veeriah, T. Zahavy, M. Hessel, Z. Xu, J. Oh, I. Kemaev, H. P. van Hasselt, D. Silver, and S. Singh, "Discovery of options via meta-learned subgoals," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[30] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," *Advances in neural information processing systems*, vol. 10, 1997.

[31] W. Ren, P. Wang, X. Li, C. E. Hughes, and Y. Fu, "Semi-supervised drifted stream learning with short lookback," in *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2022, pp. 1504–1513.

[32] J. Gehring, G. Synnaeve, A. Krause, and N. Usunier, "Hierarchical skills for efficient exploration," *Advances in Neural Information Processing Systems*, vol. 34, pp. 11 553–11 564, 2021.

[33] N. K. Jong, T. Hester, and P. Stone, "The utility of temporal abstraction in reinforcement learning." in *AAMAS (1)*. Citeseer, 2008, pp. 299–306.

[34] K. Hakhamaneshi, R. Zhao, A. Zhan, P. Abbeel, and M. Laskin, "Hierarchical few-shot imitation with skill transition models," in *International Conference on Learning Representations*, 2021.

[35] E. Jang, S. Gu, and B. Poole, "Categorical reparametrization with gumble-softmax," in *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net, 2017.

[36] O. Celik, D. Zhou, G. Li, P. Becker, and G. Neumann, "Specializing versatile skill libraries using local mixture of experts," in *Conference on Robot Learning*. PMLR, 2022, pp. 1423–1433.

[37] M. I. Belghazi, A. Baratin, S. Rajeshwar, S. Ozair, Y. Bengio, A. Courville, and D. Hjelm, "Mutual information neural estimation," in *International conference on machine learning*. PMLR, 2018, pp. 531–540.

[38] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, "Learning deep representations by mutual information estimation and maximization," in *International Conference on Learning Representations*, 2018.

[39] T. Zhao, D. Luo, X. Zhang, and S. Wang, "Towards faithful and consistent explanations for graph neural networks," in *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, 2023, pp. 634–642.

[40] J. Bekker and J. Davis, "Learning from positive and unlabeled data: A survey," *Machine Learning*, vol. 109, no. 4, pp. 719–760, 2020.

[41] T. Zhao, X. Zhang, and S. Wang, "Exploring edge disentanglement for node classification," in *Proceedings of the ACM Web Conference 2022*, 2022, pp. 1028–1036.

[42] M. Chevalier-Boisvert, L. Willems, and S. Pal, "Minimalistic gridworld environment for openai gym," https://github.com/maximecb/gym-minigrid, 2018.

[43] J. M. Bajor and T. A. Lasko, "Predicting medications from diagnostic codes with recurrent neural networks," 2016.

[44] S. Wang, S. Toyer, A. Gleave, and S. Emmons, "The `imitation` library for imitation learning and inverse reinforcement learning," https://github.com/HumanCompatibleAI/imitation, 2020.

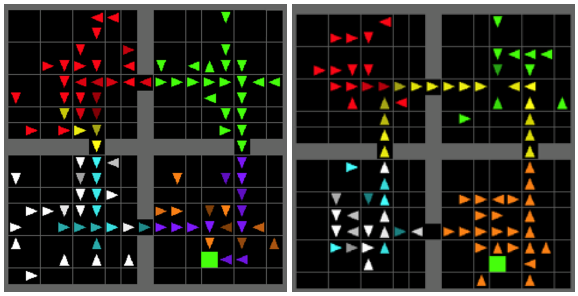
[45] B. D. Ziebart, J. A. Bagnell, and A. K. Dey, "Modeling interaction via the principle of maximum causal entropy," in *ICML*, 2010.

[46] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[47] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *ICLR (Poster)*, 2015.

[48] L. Wang, R. Tang, X. He, and X. He, "Hierarchical imitation learning via subgoal representation learning for dynamic treatment recommendation," in *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, 2022, pp. 1081–1089.

[49] M. Singer, C. S. Deutschman, C. W. Seymour, M. Shankar-Hari, D. Annane, M. Bauer, R. Bellomo, G. R. Bernard, J.-D. Chiche, C. M. Coopersmith *et al.*, "The third international consensus definitions for sepsis and septic shock (sepsis-3)," *Jama*, vol. 315, no. 8, pp. 801–810, 2016.



(a) Skill selection of $\mathcal{D}^{clean}$ (b) Skill selection of $\mathcal{D}^{noisy}$

Figure 8: Visualization of skill selection behavior on dataset FourRoom. We test the bi-directional skill selection module on demonstrations sampled from $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$ respectively, and use each color to denote a concrete skill variable. Transparency represents confidence of the selected skill.

A DATASETS

FourRoom. In FourRoom, the agent needs to navigate in a maze composed of four rooms interconnected by 4 gaps in the walls, and the goal is to reach a specific target position [42]. The expert demonstration set is constructed by collecting successful trajectories from a pre-trained agent, and the noisy demonstration set contains trajectories that fail to reach the target position (arrive at another position). In total, this dataset contains 2,000 expert demonstrations (average trajectory length: 23.42) and 2,000 noisy demonstrations (average trajectory length: 21.08).

DoorKey. This environment is a two-room maze, in which the agent must first pick up a key and then unlock the door before getting to the target square in the other room. It is a challenging environment due to the sparsity of rewards. Utilizing a pre-trained agent, the clean and noisy sets are constructed as its generated trajectories of different reward ranges. Concretely, 2,000 trajectories with rewards higher than 92 are selected as the clean set (average trajectory length: 9.71), and 2,000 trajectories with rewards within [30, 90] are used as the noisy set (average trajectory length: 9.88).

Mimic-IV. Mimic-IV contains 43,000 patients in intensive care units during the year 2001 and 2012. Following the procedures in [48], we extract the *Sepsis* patients from it conforming to the Sepsis-3 criteria [49]. For each patient, a trajectory of received medication treatments is extracted (a demonstration), with each time step corresponding to four hours that is the median of the prescription frequency in this dataset [48]. Medication action at each time step is represented as a one-hot vector with a length of 25. The observation at each time step contains relevant physiological features including both static and temporally dynamic ones, along with the historical medication actions at the previous step. This dataset contains 6,630 trajectories that the patient is fully recovered (average trajectory length: 12.92, we use them as clean demonstrations) and 3,573 trajectories that the patient's health condition deteriorates or is deceased (average trajectory length: 13.08, used as noisy demonstrations). The objective is to train a neural agent to learn an expert prescription policy from these demonstrations.

B BASELINES

In this part, we briefly introduce those baselines compared to in this work. First are those representative imitation learning and treatment recommendation methods:

- Behavior Cloning (BC) [5]: BC bins the demonstrations into transitions and learns an action-taking policy in a supervised manner.
- DensityIRL [44]: Density-based reward modeling conducts a non-parametric estimation of marginal as well as joint distribution of states and actions from provided demonstrations, and computes a reward using the log of those probabilities.
- MCE IRL [45]: Its full name is Maximum causal entropy inverse reinforcement learning, whose idea is similar to DensityIRL. It designs an entropy-based approach to estimate causally conditioned probabilities for the estimation of reward function.
- AIRL [6]: Adversarial inverse reinforcement learning aims to automatically acquiring a scalable reward function from expert demonstrations by adversarial learning.
- GAIL [7]: GAIL directly trains a discriminator adversarially to provide the reward signal for policy learning.

The above methods are all proposed assuming demonstrations to be clean and optimal. We further compare SDIL with two recent strategies that are designed to utilize noisy demonstrations explicitly: ACIL [15] and DWBC [9].

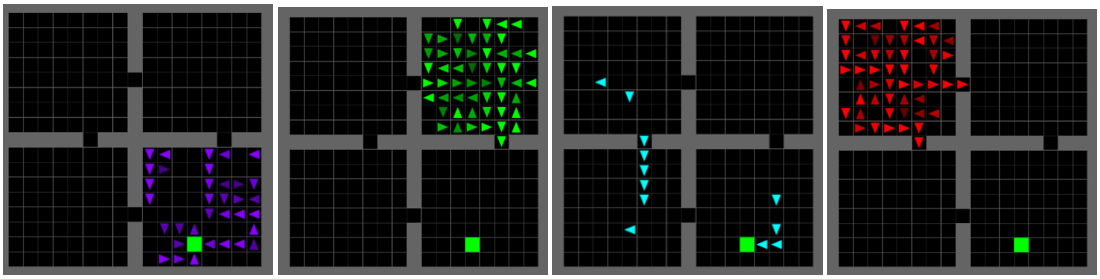
- ACIL [15]: ACIL extends GAIL by incorporating noisy demonstrations as negative examples and learns the policy model to be distinct from them.
- DWBC [9]: It trains a discriminator to evaluate the quality of each demonstration trajectory through adversarial training, and conducts a weighted behavior cloning.

C CASE STUDY

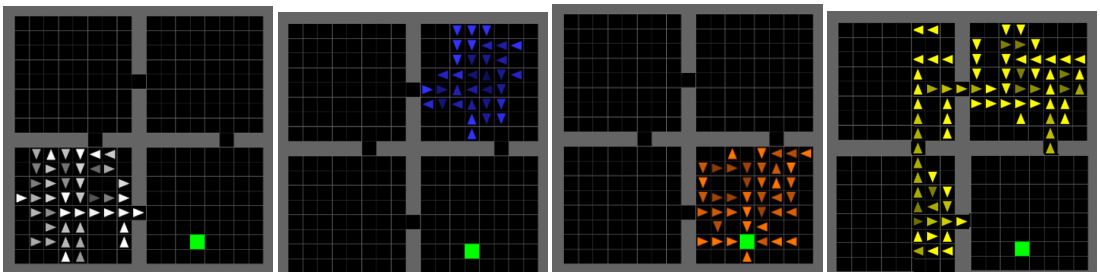
FourRoom is a gym environment of four rooms in which the agent needs to navigate and reach the target grid. In this environment, behaviors of the agent are easy to interpret which allows us to directly visualize and evaluate learned skills. To interprete knowledge learned by SDIL, we conduct two sets of studies: 1) distribution of skill selections along clean and noisy demonstrations; 2) action-taking strategies encoded by each concrete skill.

C.1 Skill Selection Behaviors

First, we analyze skill selection behaviors of the expert demonstration set, and compare them with skill selection behaviors of the noisy demonstration set in Fig. 8. Concretely, we test the skill selection behavior of SDIL on given demonstrations for a case study. 50 demonstrations are selected from $\mathcal{D}^{clean}$ and $\mathcal{D}^{noisy}$ respectively, and we visualize them in the FourRoom environment as in Fig. 8. Arrow direction represents the majority moving direction of sampled demonstrations at that position, and those that have a high diversity in action selection are discarded. We annotate the selected skills using different colors, and its transparency shows the consistency of skill selection across demonstrations at that position. From the figure, it can be observed that each skill encodes a particular moving strategy inside a region, and some skills are shared by both demonstration sets like the red one. There are also some skills encoding moving strategy specific for one demonstration set, like the purple one in Fig. 8(a) and the yellow one in Fig. 8(b).



(a) Skill with $1st$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$ (b) Skill with $2nd$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$ (c) Skill with $3rd$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$ (d) Skill with $4th$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$



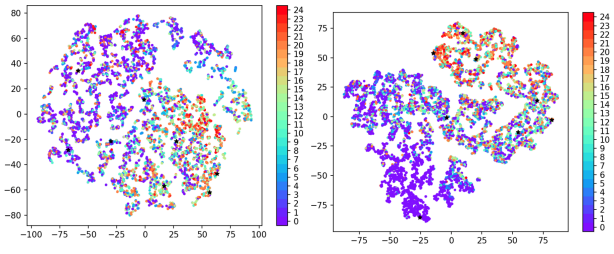
(e) Skill with $5th$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$ (f) Skill with $6th$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$ (g) Skill with $7th$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$ (h) Skill with $8th$ preference score by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$

Figure 9: Visualization of learned skills from dataset FourRoom, after the skill discovery phase. We rank them through comparing their probability of being selected by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$. For each skill, we randomly run the agent to generate 50 trajectories with the skill variable fixed. We only visualize the transitions when the pre-fixed skill has the highest probability of being selected to better highlight its encoded behaviors. Arrow at each grid denotes the moving direction, and its transparency represents confidence of the selected action.

C.2 Behavior Learned by Skills

FourRoom Here, we visualize the learned skills in Fig. 9. For each skill, we force SDIL to generate 50 trajectories with the skill variable fixed. For ease of interpretation, we only visualize the regions in which this skill has a high probability of being selected itself. Arrow direction indicates the majority moving direction, and its transparency shows its consistency. We rank the skills based on their preference by $\mathcal{D}^{clean}$ over $\mathcal{D}^{noisy}$. It is clear to see that: 1) Each skill represents a particular moving strategy within a region. For example, Fig. 9(a) shows the skill of heading to the target region inside the same room, and Fig. 9(c) corresponds to the behavior of going across a pathway in a particular direction. 2) Our SDIL is able to discover separate skills for different strategies of the same region, like comparing Fig. 9(b) to Fig. 9(f). Skill in Fig. 9(b) targets to move to the lower-right room, while that in Fig. 9(f) would stay in this room and head to a particular position.

Mimic-IV In the medical dataset Mimic-IV, it is difficult to directly interpret learned skills, as each skill would encode a particular distribution of actions conditioned on observed states. Faced with these difficulties, we conduct the case study from another perspective: by analyzing the distribution of latent embeddings w.r.t skill variables and target actions. Concretely, we draw the TSNE visualization in Fig. 10. Each node represents the embedding of a transition extracted by the skill encoding module, black pentagrams denote the embedding of skill variables, and the color represents the selected treatments. Note that in Mimic-IV dataset, a larger treatment label corresponds to a more active medical plan. To further make the comparison, we test two variants of SDIL with the weight of mutual information regularization being set to 1 and 0 respectively. Results are shown in Fig. 10. Comparing two figures,



(a) $\lambda = 0$ (b) $\lambda = 1$

Figure 10: TSNE Visualization of skill distribution of dataset Mimic-IV. Each color represents a different treatment action, and embedding of skill variables are marked with a black pentagram. In our data split, severity of health conditions would generally increase with a larger treatment class.

it can be observed that skills in Fig. 10(b) has a stronger correlation with treatment distributions. Transitions taking similar treatments are clustered better compared to Fig. 10(b).