Learning Strategy Representation for Imitation Learning in Multi-Agent Games

Shiqi Lei[1*], Kanghoon Lee[2*], Linjing Li[1,3], Jinkyoo Park[2†]

[1]Institute of Automation, Chinese Academy of Sciences (CASIA)
[2]Korea Advanced Institute of Science and Technology (KAIST)
[3]Beijing Wenge Technology Co., Ltd.
leishiqi2022@ia.ac.cn, leehoon@kaist.ac.kr, linjing.li@ia.ac.cn, jinkyoo.park@kaist.ac.kr

Abstract

The offline datasets for imitation learning (IL) in multi-agent games typically contain player trajectories exhibiting diverse strategies, which necessitate measures to prevent learning algorithms from acquiring undesirable behaviors. Learning representations for these trajectories is an effective approach to depicting the strategies employed by each demonstrator. However, existing learning strategies often require player identification or rely on strong assumptions, which are not appropriate for multi-agent games. Therefore, in this paper, we introduce the Strategy Representation for Imitation Learning (STRIL) framework, which (1) effectively learns strategy representations in multi-agent games, (2) estimates proposed indicators based on these representations, and (3) filters out sub-optimal data using the indicators. STRIL is a plug-in method that can be integrated into existing IL algorithms. We demonstrate the effectiveness of STRIL across competitive multi-agent scenarios, including Two-player Pong, Limit Texas Hold'em, and Connect Four. Our approach successfully acquires strategy representations and indicators, thereby identifying dominant trajectories and significantly enhancing existing IL performance across these environments.

Introduction

Although reinforcement learning has become a powerful technique for sequential decision-making in various domains such as robotic manipulation (Andrychowicz et al. 2020), autonomous driving (Chen, Yuan, and Tomizuka 2019), and game playing (Vinyals et al. 2019), conventional reinforcement learning demands substantial online interactions with the environment, which can be costly and sample inefficient while potentially leading to safety risks (Berner et al. 2019; Bojarski et al. 2016). To address these issues, many methods have emerged to enable efficient learning using offline datasets generated by demonstrators. For example, imitation learning (IL) (Pomerleau 1988) replicates actions from the offline dataset without reward information, while offline reinforcement learning (Fujimoto, Meger, and Precup 2019; Kumar et al. 2020) is provided access to reward signals. Offline learning datasets are usually collected from multiple demonstrators to enlarge dataset scale and diversity (Sharma et al. 2018; Mandlekar et al. 2019, 2021), which leads to a dataset of behaviors with various characteristics. However, standard IL algorithms treat all data samples in the dataset as homogeneous, potentially learning undesired behaviors from sub-optimal trajectories.

To address the above issue, the key insight in our proposed method is to assign each trajectory in the offline dataset with a unique learned attribute, i.e., strategy representation, so that we can further analyze each trajectory considering its specificity and filter out sub-optimal data. With a precise depiction of each trajectory and their distribution on the representation space, we can judge the performance of each trajectory by only collecting a few (less than 5%) data with trajectory rewards or even without any reward information. In this work, we introduce Strategy Representation for Imitation Learning (STRIL), an efficient and interpretable approach designed to improve IL by filtering sub-optimal demonstrations from offline datasets.

Figure 1 illustrates an overview of STRIL. Note that STRIL is a plug-in method compatible with existing IL algorithms. It consists of three components: strategy representation learning using a Partially-trainable-conditioned Variational Recurrent Neural Network (P-VRNN), indicator estimation, and filtered IL. The detailed steps and corresponding contributions are outlined as follows:

- We propose an unsupervised framework with P-VRNN to efficiently extract strategy representations from multi-agent game trajectories. Strategy representation for each trajectory is customized as a network condition.
- We define the Randomness Indicator (RI) and Exploited Level (EL), which utilize strategy representation to effectively evaluate offline trajectories in a zero-sum game. EL can be precisely estimated even with limited reward data, while RI requires no reward data.
- We enhance existing IL methods by filtering out sub-optimal trajectories using the RI and EL indicators, ensuring that IL is trained only on the dominant trajectory.
- We demonstrate that STRIL can provide effective strategy learning without player identification and significantly enhance the performance of various IL algorithms in competitive zero-sum games, including Two-player Pong, Limit Texas Hold'em, and Connect Four.

*Equal contribution
†Corresponding author

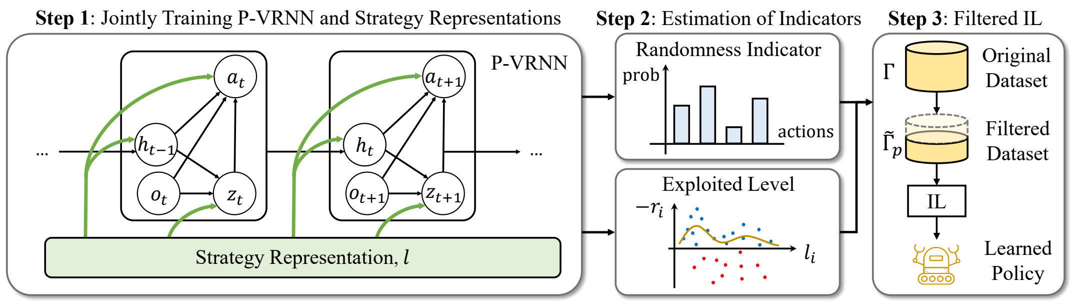


Figure 1: The overall diagram of Strategy Representation for Imitation Learning (STRIL).

Related Works

Imitation Learning. In the conventional IL settings, the expert trajectories only have information of state-action pairs without reward information (Pomerleau 1988; Ross, Gordon, and Bagnell 2011; Ho and Ermon 2016; Ding et al. 2019; Garg et al. 2021), and it is assumed that the demonstrations are homogeneous oracle. However, realistic crowd-sourced datasets are usually multi-modal and include sub-optimal demonstrations. Some IL methods are proposed for multi-modal offline datasets, such as (Hausman et al. 2017) and (Fei et al. 2020). As for sub-optimal data, there are plenty of approaches to alleviate the negative influence (Brown, Goo, and Niekum 2020; Chen, Paleja, and Gombolay 2021; Zhang et al. 2021; Kim et al. 2022; Xu et al. 2022), but all these methods require environment dynamics, the rankings over the demonstrations, or the identification of demonstrators. In contrast, our method does not require such information. Sasaki and Yamashina (2020) enhances behavior cloning (BC) with noisy demonstrations, but their method does not deal with general sub-optimal trajectories. TRAIL (Yang, Levine, and Nachum 2021) achieves sample-efficient IL via a learned latent action space and a factored transition model. We would like to additionally mention the work by Franzmeyer et al. (2024), which adopts a similar framework of filtering the offline dataset and uses an IL algorithm. Nevertheless, their method assumes a cooperative setting and requires reward information.

IL with Representation Learning. The work by Beliaev et al. (2022) closely aligns with our research, sharing the primary goal of extracting expertise levels of trajectories. They assume that the demonstrator has a vector indicating the expertise of latent skills, with each skill requiring a different level at a specific state. These elements jointly derive the expertise level. The method also considers the policy worse when it is closer to uniformly random distribution. However, this assumption cannot be satisfied even in simple games like RPS, where a uniformly random strategy constitutes a Nash equilibrium. Play-LMP (Lynch et al. 2020) leverages unsupervised representation learning in a latent plan space for improved task generalization. However, employing a variational auto-encoder (VAE) with the encoder outputting latent plans is unsuitable for multi-player games, potentially leaking opponent information from the observations and disrupting the evaluation of the demonstrator. Grover et al. (2018) also studies learning policy representations from offline trajectories. They use the information of agent identification during training, which enables them to add a loss to distinguish one agent from others. However, this information may not be provided in the offline datasets.

Preliminaries

Markov Games. A Markov game (Littman 1994) is a partially observable Markov decision process (Kaelbling, Littman, and Cassandra 1998) (POMDP) adapted to a multi-agent setting, where each agent has its own reward function. In a Markov game, there is a state space S and n agents, with each agent i having a corresponding action space A_i and observation space O_i. When an agent is not required to take action at a certain state, its action space contains only one action, referred to as a 'null' action. At each time step t: (1) each agent i obtains an observation $o_t^i \in O_i$ and selects an action $a_t^i \in A_i$ based on the policy of agent i, $\pi_i : O_i \times A_i \to [0, 1]$; (2) the agent receives a reward $\mathfrak{r}_t^i : S \times A_i \to \mathbb{R}$ based on the state and the action; (3) the state is changed according to the transition function $T : S \times A_1 \times ... \times A_n \to S$. For a complete trajectory $\tau = ((o_0^i, a_0^i), ..., (o_T^i, a_T^i))$ of agent i, there is a reward $\mathfrak{r}_t^i$ received at each time step t. We define the trajectory reward τ as $\hat{r}_i(\tau) = \sum_{t=0}^{T} \mathfrak{r}_t^i$. The expected trajectory reward of player i with strategy π_i and opponents with strategy π_{-i} is defined as $r_i(\pi_{-i}, \pi_i) = \mathbb{E}_{\tau\sim(\pi_{-i},\pi_i)} [\hat{r}_i(\tau)]$, in which $\tau \sim (\pi_{-i}, \pi_i)$ denote that τ is generated with agents using strategy (π_{-i}, π_i).

Best Response, Exploitability, and Nash Equilibrium. We use $r_i(\pi_{-i}, \pi_i)$ to specify the reward of the player playing π_i against π_{-i}. The best response of opponent strategy π_{-i} is defined as $BR(\pi_{-i}) = \text{argmax}_{\pi_i'} r_i(\pi_{-i}, \pi_i')$, which refers to the strategy of player i that maximizes player i's reward. We additionally define the best response of strategy π_i as $BR(\pi_i) = \text{argmax}_{\pi_{-i}'} \sum_{j\in P, j\neq i} r_j(\pi_{-i}', \pi_i)$, which equals to $\text{argmin}_{\pi_{-i}'} r_i(\pi_{-i}', \pi_i)$ in the zero-sum case. $BR(\pi_i)$ refers to the strategy of all the other players except player i that maximizes their trajectory reward, which is equivalent to minimizing player i's payoff in zero-sum games. Let P be the set of all the players in the game, and the strategy $\pi = (\pi_i)_{i\in P}$ be the strategy of all the players. We define the exploitability of strategy π as $E(\pi) = \sum_{i\in P} (r_i(\pi_{-i}, BR(\pi_{-i})) - r_i(\pi_{-i}, \pi_i))$, which reflects the extent to which the strategy can be exploited. In zero-sum symmetric cases, we define the exploitability of a player strategy π_i as $E(\pi_i) = -r_i(BR(\pi_i), \pi_i) =$

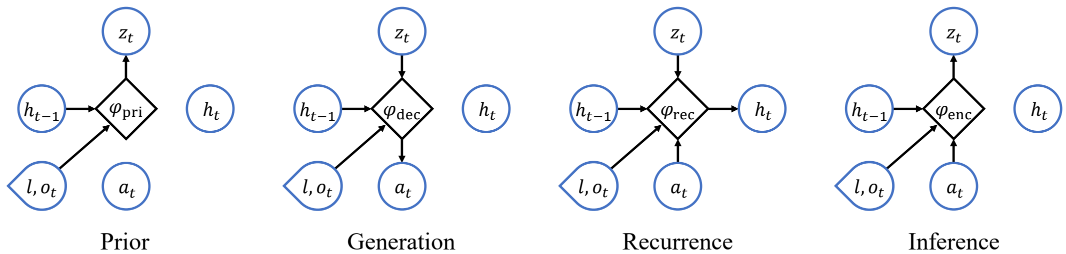


Figure 2: The decomposed network structure of the P-VRNN model. The variables are depicted as circles, learnable parameters as diamonds, and partially-trainable variables as a combination of both diamonds and circles.

$\sum_{j \in P, j \neq i} r_j(BR(\pi_i), \pi_i)$. A strategy π_i is ε-Nash equilibrium if $E(\pi_i) \leq \varepsilon$.

Problem Formulation

Consider a multi-player competitive zero-sum game, and we have a dataset of game histories that include the trajectories of each player. The trajectories are generated by diverse players, ranging from high-level experts to amateurs. We aim to extract strategy representations from trajectories, distinguish the players with different levels, and learn an expert policy from the dataset via imitation learning. We assume that we do not have the identifications of the players. In our problem, we collect a set Γ of trajectories $\tau = ((o_0, a_0), ..., (o_T, a_T))$ from different games and demonstrators. The trajectory reward for a subset $\Gamma' \subset \Gamma$ is available for exploited level estimation. We assume that the strategy of a player is consistent within a single trajectory.

Learning Strategy Representation

Identifying the strategy of a player is essential to evaluating their skill level. However, this becomes challenging when player identification is unavailable in the dataset because the strategy of the player changes according to their opponent within each episode. Therefore, we propose a Partially-trainable-conditioned Variational Recurrent Neural Network (P-VRNN) featuring a strategy representation that is learnable and remains constant throughout the trajectory. Strategy representation becomes the optimal representation for each trajectory by training it to minimize the P-VRNN loss.

The P-VRNN models the player's decision-making process and includes four major components similar to the original VRNN, as shown in Figure 2. To disentangle the strategy of the opponent player from the strategy representation, we consider the observation as a conditional variable. We define p as the generative model and q as the inference model.

Generation

Based on the dependency of our P-VRNN model, We can model the decision-making process as follows:

$$p(a_{\leq T}, z_{\leq T} | o_{\leq T}, l) = \prod_{t=1}^{T} \underbrace{p(a_t | z_{\leq t}, a_{<t}, o_{\leq t}, l)}_{\text{generation}} \underbrace{p(z_t | a_{<t}, z_{<t}, o_{\leq t}, l)}_{\text{prior}}, \quad (1)$$

Without knowing the action a_t, the prior distribution of latent variable z_t can be derived from the past actions $a_{<t}$, past latent variables $z_{<t}$, observations $o_{\leq t}$, and strategy representation l. The computation graph of the P-VRNN shows that the recurrent variable h_{t-1} integrates the past actions $a_{<t}$, latent variables $z_{<t}$, and observations $o_{<t}$. Therefore, with the assumption of a Gaussian distribution for the prior, the sampling process of z_t is influenced by h_{t-1}, the current observation o_t, and the strategy representation l as follows:

$$\begin{aligned} z_t \mid h_{t-1}, o_t, l &\sim \mathcal{N}(\mu_{\text{pri},t}, \text{diag}(\sigma^2_{\text{pri},t})), \\ [\mu_{\text{pri},t}, \sigma_{\text{pri},t}] &= \varphi_{\text{pri}}(h_{t-1}, o_t, l), \end{aligned} \quad (2)$$

where φ_{pri} is a prior network. We also follow the convention in VAE and assume that the latent variable has a diagonal covariance matrix.

The generation process is obtaining action a_t from the latent variables $z_{\leq t}$, past actions $a_{<t}$, observations $o_{\leq t}$, and strategy representation l just same as to the decision-making process of the player. By substituting the past information using recurrent variable h_{t-1}, action generation is defined as follows:

$$\begin{aligned} a_t \mid h_{t-1}, z_t, o_t, l &\sim \text{Cat}(\mu_{\text{dec},t}), \\ \mu_{\text{dec},t} &= \varphi_{\text{dec}}(h_{t-1}, z_t, o_t, l), \end{aligned} \quad (3)$$

where Cat stands for categorical distribution and φ_{dec} is a decoder network.

The recurrent unit takes in all the variables of the current step and the recurrent variable of the previous step, which includes all the past information. At each time step, h_t is updated as follows:

$$h_t = \varphi_{\text{rec}}(h_{t-1}, a_t, z_t, o_t, l), \quad (4)$$

where φ_{rec} is a recurrent network.

Inference

Approximate posterior inference is modeled as follows:

$$q(z_{\leq T} | a_{\leq T}, o_{\leq T}, l) = \prod_{t=1}^{T} \underbrace{q(z_t | a_{\leq t}, z_{<t}, o_{\leq t}, l)}_{\text{inference}}. \quad (5)$$

The latent variable z_t is obtained from the actions $a_{\leq t}$, past latent variables $z_{<t}$, observations $o_{\leq t}$, and strategy representation l. Like the prior and action generation, we replace the past information with a recurrent variable of the previous

step, h_{t-1}. Therefore, the approximate posterior distribution is defined as follows:

$$\begin{aligned} & z_t \mid h_{t-1}, a_t, o_t, l \sim \mathcal{N}(\mu_{\text{enc},t}, \operatorname{diag}((\sigma_{\text{enc},t})^2)), \\ & [\mu_{\text{enc},t}, \sigma_{\text{enc},t}] = \varphi_{\text{enc}}(h_{t-1}, a_t, o_t, l), \end{aligned} \tag{6}$$

where φ_{enc} is a encoder network.

Learning

Similar to (Chung et al. 2015), the loss function of P-VRNN is a negative of the variational lower bound, using Equations (1) and (5), as follows:

$$\mathcal{L} = \mathbb{E}_{q_\phi(z_{\leq T} | a_{\leq T}, o_{\leq T}, l)} \left[\sum_{t=1}^{T} (\mathcal{L}_{\text{Recon},t} + \mathcal{L}_{\text{Reg},t}) \right]. \tag{7}$$

The reconstruction loss for each timestep, which evaluates how well the generated action aligns with the original action, is formulated as follows:

$$\mathcal{L}_{\text{Recon},t} = -\log p_\theta(a_t | z_{\leq t}, a_{<t}, o_{\leq t}, l). \tag{8}$$

The regularization loss for each timestep, which measures the divergence between the posterior and prior distributions, is formulated as follows:

$$\begin{aligned} & \mathcal{L}_{\text{Reg},t} \\ & = \mathrm{KL}(q_\phi(z_t | a_{\leq t}, z_{<t}, o_{\leq t}, l) \,\|\, p_\theta(z_t | a_{<t}, z_{<t}, o_{\leq t}, l)) \end{aligned} \tag{9}$$

At the beginning of the training, the strategy representation l, which is a trainable variable, is randomly initialized for each trajectory τ. The condition part of P-VRNN consists of an observation o_t that changes over time and the strategy representation l, which is consistent during the whole trajectory and trainable. During the training, all the l are optimized together with the parameters of φ_{pri}, φ_{enc}, φ_{dec}, and φ_{rec} to minimize the loss function described in Equation (7). While the networks are trained to minimize the loss across all trajectories, the strategy representations are individually optimized for each trajectory to provide customized guidance and insights. Consequently, the strategy representation l should be adjusted to more effectively capture and express the unique strategies of each trajectory. It is important to note that the process of deriving l is conducted unsupervised, without needing player identification, ensuring privacy and generalizability.

Indicators for Imitation Learning

Utilizing the learned P-VRNN and the strategy representation for each trajectory, we propose the RI and EL indicators.

Randomness Indicator (RI)

Given the well-trained P-VRNN and the strategy representation dataset, we can evaluate the reconstruction loss and regularization loss for each trajectory. The regularization loss shows the capability of the posterior to approximate the prior, which reflects the performance of extracting the information of the next action from the past information, observation, and strategy representation. In the process of P-VRNN training, the regularization loss of each trajectory is gradually optimized to a very small value close to 0. However, the reconstruction loss typically cannot be so small since the action decoder gives a probability distribution over actions, and players usually do not act deterministically. For a well-trained P-VRNN, the predicted action distribution closely matches the true probability distribution of the corresponding strategy of the trajectory. So if there are n possible actions $a_{t,1}, a_{t,2}, ..., a_{t,n_t}$ for a_t, we can approximately calculate the expectation of the one-step reconstruction loss as

$$\begin{aligned} & \mathbb{E}_{p_\theta(a_t | z_{\leq t}, a_{<t}, o_{\leq t}, l)} [\mathcal{L}_{\text{Recon},t}] \\ & = \sum_{i=1}^{n_t} -p_\theta(a_{t,i} | z_{\leq t}, a_{<t}, o_{\leq t}, l) \log p_\theta(a_{t,i} | z_{\leq t}, a_{<t}, o_{\leq t}, l) \\ & = \mathrm{H}\left(p_\theta(a_t | z_{\leq t}, a_{<t}, o_{\leq t}, l)\right). \end{aligned} \tag{10}$$

It is the entropy of $p_\theta(a_t | z_{\leq t}, a_{<t}, o_{\leq t}, l)$, which reflects the randomness of the player with strategy representation l, given $z_{\leq t}, a_{<t}$, and $o_{\leq t}$. Following the hypothesis of Beliaev et al. (2022), a strategy with more randomness is considered worse. Since we have the whole trajectory with a unified strategy representation l, we can define the RI of a trajectory as its cumulative reconstruction loss:

$$RI(\tau) = \sum_{t=1}^{T} \mathrm{H}\left(p_\theta(a_t | z_{\leq t}, a_{<t}, o_{\leq t}, l)\right). \tag{11}$$

We highlight that the RI does not require any reward information, and the whole procedure is fully unsupervised.

Exploited Level (EL)

If we can access the trajectory rewards of select trajectories, we can determine to what extent the strategy of each trajectory in the offline dataset is exploited by utilizing the geometric structure of the strategy representation space. The key insight of this approach is that the trajectories with similar strategy representations tend to exhibit similar strategies.

We define measure $\mathrm{d}\pi$ on the strategy space Π, and assume that the strategy of agents generating dataset Γ and its subset Γ' are both sampled according to $\mathrm{d}\pi$. Denote a trajectory as τ and the representation function mapping trajectories to learned representations as $f(\tau)$. We remark that a trajectory τ should be mapped to a probability distribution of strategies such that $\int_{\pi \in \Pi} \tau(\pi) \mathrm{d}\pi = 1$, where $\tau(\pi)$ is the probability of using strategy π when having trajectory τ, instead of a single strategy. But we can view the mixture of π with probability $\tau(\pi)$ as a single mixed strategy $\int_{\pi \in \Pi} \pi \tau(\pi) \mathrm{d}\pi$, so we can still use notation $\pi(\tau)$ to represent the strategy of τ. We define the EL as follows:

$$EL(\tau) = \mathbb{E}_\pi \left[-r(\pi, \pi(\tau)) \mid r(\pi, \pi(\tau)) \leq 0 \right] \tag{12}$$

$$= \frac{\int_{\pi \in \Pi} (-r(\pi, \pi(\tau))^+ \mathrm{d}\pi}{\int_{\pi \in \Pi} \mathbb{1}_{r(\pi, \pi(\tau)) \leq 0} \mathrm{d}\pi}, \tag{13}$$

where $r(\pi, \pi(\tau))$ returns the expected trajectory reward of a player with strategy $\pi(\tau)$ by default, $(x)^+ = \max\{x, 0\}$, and $\mathbb{1}_c = 1$ if and only if condition c is satisfied, otherwise $\mathbb{1}_c = 0$. $EL(\tau)$ is the negative of the expectation of the trajectory reward less than 0 when played with the demonstrators who generate the offline dataset. This value can reflect the extent to which the demonstrators exploit the strategy of τ. To estimate EL with latent representation space structure, we provide an alternative definition of EL_δ:

$$EL_\delta(\tau) = \frac{\sum_{d(f(\tau),f(\tau'))<\delta}(-r(\hat{\pi},\pi(\tau')))^+}{\sum_{d(f(\tau),f(\tau'))<\delta}\mathbb{1}_{r(\hat{\pi},\pi(\tau'))\leq 0}}, \quad (14)$$

where d is a metric over the strategy representation space. Due to the Lipschitz continuity of the P-VRNN with respect to the representation, the trajectories with similar strategy representations have similar strategies. Thus, to approximate the negative EL of τ, we can calculate the mean of all the negative rewards of the trajectories with the strategy representations in the small neighborhood of τ's representation. It can be proved that $\lim_{\delta\to 0^+} EL_\delta(\tau) = EL(\tau)$. EL_δ has favorable properties, such as the low value near Nash equilibrium strategies. Given a trajectory τ and its corresponding distribution $\tau(\pi)$ over Π, $\pi(\tau)$ is ϵ_1-Nash equilibrium, and we assume that any pure strategy can exploit another strategy by at most M. We also assume that similar representations induce similar strategies: if $d(f(\tau_1), f(\tau_2)) < \delta$, then $\int_{\pi\in\Pi} |\tau_1(\pi) - \tau_2(\pi)| \mathrm{d}\pi < \alpha\delta$, where α is a constant. It can be proved that $EL_\delta(\tau) < \epsilon_1 + \alpha\delta M$.

Since EL is the average of values satisfying conditions with distance constraints on the representation space, we can train an operator L to estimate EL from representation. We have the representation l and trajectory reward $\hat{r}$ for each trajectory τ, and we intend to minimize $\sum_{\hat{r}^i\geq 0} ||L(l^i) - \hat{r}^i||_2$, where l^i and $\hat{r}^i$ are the representation and trajectory reward of the i-th trajectory τ^i in the dataset, so that the prediction from $L(l)$ becomes close to the mean of satisfying reward $\hat{r} \geq 0$ nearby. We use a two-layer MLP as L. After training L, we can directly obtain the EL of a single trajectory τ even without the reward information. By applying EL estimator L on the representation $f(\tau)$ of trajectory τ, we can get the desired result $L(f(\tau))$.

Filtered Imitation Learning

The last step of the STRIL is to filter the offline dataset with a chosen percentile p of an indicator. The indicator I can be any mapping from the trajectories to real numbers such as RI or EL. Specifically, for an indicator $I(\tau)$, the offline dataset Γ is filtered into $\tilde{\Gamma}_p = \{\tau \in \Gamma \mid I(\tau) < I_p\}$, where I_p satisfies that $\mathbb{P}_\tau[I(\tau) < I_p] = p$. After filtering the dataset, the original IL algorithm is employed. For IL algorithms that directly define loss function over target function and trajectories, the new loss function can be explicitly written as

$$\mathcal{L}_p(\pi) = \mathbb{E}_\tau\left[\mathbb{1}_{I(\tau)<I_p} \cdot \mathcal{L}^{\mathrm{IL}}(\pi,\tau)\right], \quad (15)$$

where $\mathcal{L}^{\mathrm{IL}}(\pi,\tau)$ is the loss function of the IL algorithm. As an example, $\mathcal{L}^{\mathrm{IL}}(\pi,\tau) = \sum_{t=0}^{|\tau|} \log \pi(a_t \mid o_t)$ in vanilla BC algorithm. As the value of p closer to 0, more data is filtered out; conversely, setting p to 1 filters none of the data, reverting STRIL to the original IL algorithm.

Experiments

Experiment Settings

We validate our approach using two-player zero-sum games: Two-player Pong, Limit Texas Hold'em (Zha et al. 2020), and Connect Four (Terry et al. 2021).

Dataset generation. We employ different methods to create training datasets with diverse demonstrators for the environments. For Two-player Pong and Connect Four, we use self-play with opponent sampling (Bansal et al. 2018) with the Proximal Policy Optimization (PPO) algorithm (Schulman et al. 2017). For Limit Texas Hold'em, we use neural fictitious self-play (Heinrich and Silver 2016) with Deep Q-network (DQN) algorithm (Mnih et al. 2013) to generate expert policies, given its complexity and the need to adapt to various opponents. Behavior models are then selected from multiple intermediate checkpoints to generate the offline data. We assume that only 5% of the dataset is reward-labeled for EL estimation.

Evaluation metrics. We evaluate our method across three environments to demonstrate the effectiveness of the learned strategy representation in STRIL using estimated indicators. In a zero-sum game, evaluating policy performance involves ensuring the policy is not vulnerable to exploitation by a specific strategy. In order to capture the worst-case scenario against opponent strategies in the dataset, we evaluate the performance of the imitative model, π_i, using the Worst Score (WS) over the demonstrator set, $\mathcal{I}$:

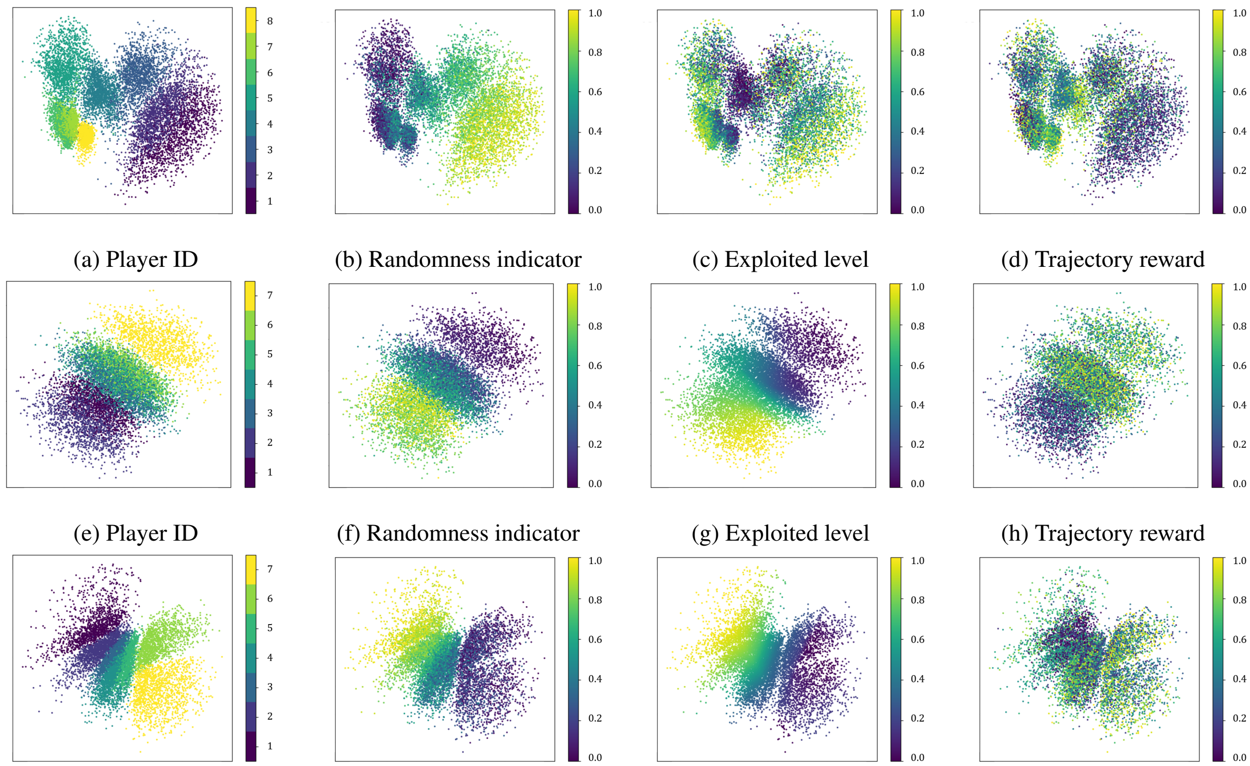
$$\mathrm{WS}(\mathcal{I}, \pi_i) = \min_{j\in\mathcal{I}} r_i(\pi_j, \pi_i), \quad (16)$$

where $r_i(\pi_j, \pi_i)$ represents the trajectory reward of i against j. For Two-player Pong and Connect Four, we calculate the reward using the formula $(N_{\text{win}} - N_{\text{lose}})/N_{\text{game}}$, where N_{win}, N_{lose}, and N_{game} represent the number of wins, losses, and total games, respectively. For Limit Texas Hold'em, where a player can win by varying margins depending on the game, we determine the reward as the average difference between the total chips won and lost. We set N_{game} to 2,000.

Strategy Representation with Indicators

In this subsection, we visualize the learned strategy representations using multiple labels. If the latent space exceeds two dimensions, it is initially reduced to two dimensions using PCA. These reduced representations are then color-coded based on different labels: player ID, RI, EL, and trajectory reward. Note that player ID and trajectory reward serve as ground truth references while RI and EL are estimated. Instead of using the exact values, we color the percentiles of RI, EL, and trajectory reward.

Two-player Pong. As shown in Figure 3a, strategy representations of each demonstrator naturally cluster together in the Two-player Pong environment. Figure 3d demonstrates that trajectory rewards only partially align with player strategies due to performance variability depending on the opponent's strategy, a common characteristic of competitive games. Players 1 and 8, exhibiting the worst and best performances, respectively, form clusters with consistent values independent of the opponent. Figure 3b illustrates that the RI highlights players 5, 6, and 8 as excelling in reconstruction tasks. Additionally, Figure 3c shows that the dominant players, 4 and 8, have strategies that are least susceptible to exploitation, signifying more robust performance. Additionally, it is observed that the most expansive cluster with the



(a) Player ID (b) Randomness indicator (c) Exploited level (d) Trajectory reward

(e) Player ID (f) Randomness indicator (g) Exploited level (h) Trajectory reward

(i) Player ID (j) Randomness Indicator (k) Exploited Level (l) Trajectory reward

Figure 3: The learned strategy representations with different labels on the Two-player Pong (a-d), Limit Texas Hold'em (e-h), and Connect Four (i-l) environments.

lowest density has the highest EL, suggesting that the least trained strategy exhibits unstable behavior and is the most vulnerable one to exploitation. These indicators establish a strong standard for data filtering in subsequent IL applications from two different perspectives, both differentiating between dominant and dominated strategies.

Limit Texas Hold'em & Connect Four. In Limit Texas Hold'em, there are seven players: two experts, three mid-level players, and two novice players. Figure 3e demonstrates that the learned representations are well separated and ordered according to their expertise levels. Figure 3h illustrates that while trajectory rewards can effectively identify very poor strategies, they fail to consistently differentiate among more effective strategies, as the rewards vary across different opponents. However, Figures 3f and 3g show that our proposed indicators not only perfectly distinguish the dominant strategies but also rank them accurately. In Connect Four, Figure 3i shows dominant player strategies on the right and dominated player strategies on the left. In contrast to the trajectory rewards which are inconsistent within the same strategy, our RI and EL patterns show a strong capability to extract characteristics and assess the performance of these strategies.

Learning from Offline Dataset

To evaluate the STRIL, we considered three IL algorithms. First, we employed BC, a basic IL algorithm. Next, we used IQ-Learn (Garg et al. 2021), an advanced imitative algorithm. Finally, we implemented ILEED (Beliaev et al. 2022), a state-of-the-art method capable of handling a diverse range of demonstrator data. In our evaluation, we excluded methods that rely on online interactions (e.g., GAIL (Ho and Ermon 2016)) or necessitate interactions with experts (e.g., DAgger (Ross, Gordon, and Bagnell 2011)) in offline learning approaches. We applied STRIL to each algorithm to evaluate its performance enhancement. Note that all the experiments were repeated three times.

General results. In Table 1, we compared the WS of four types of data filtering methods. A hyperparameter search was conducted to identify the appropriate percentile, p, of indicators for each model and environment. Note that all experiments were repeated three times, and the results are reported with error bars. The original ILEED, which considers the expertise level of the data, generally performs better than other original algorithms on average. For the filtering method, the RI and EL enhance the performance of the original methods in most cases. In some instances, their performance is even comparable to the Best method. In the case of Two-player Pong, the EL method outperforms the RI method because EL more accurately distinguishes the dominant strategy. Additionally, in Limit Texas Hold'em, both RI and EL show similar performance, which aligns with the similar qualitative results observed in the strategy space. However, the RI and EL methods did not improve ILEED on Connect Four because the dataset aligns well with the assumption of ILEED. Consequently, filtering the data ac-

Game	Algorithm	Filtering Method			
		Original	RI	EL	Best
Two-Player Pong	BC	-0.832 ± 0.011	-0.613 ± 0.052	$\mathbf{-0.343 \pm 0.036}$	-0.044 ± 0.033
	IQ-Learn	-0.804 ± 0.044	-0.601 ± 0.008	$\mathbf{-0.254 \pm 0.063}$	-0.009 ± 0.013
	ILEED	-0.711 ± 0.070	-0.607 ± 0.058	$\mathbf{-0.458 \pm 0.118}$	-0.031 ± 0.016
Limit Texas Hold'em	BC	-1.255 ± 0.123	0.532 ± 0.052	$\mathbf{0.662 \pm 0.011}$	0.464 ± 0.103
	IQ-Learn	-3.652 ± 0.428	$\mathbf{0.667 \pm 0.097}$	0.618 ± 0.027	0.640 ± 0.061
	ILEED	-0.411 ± 0.150	$\mathbf{0.654 \pm 0.033}$	0.494 ± 0.065	0.487 ± 0.058
Connect Four	BC	-0.353 ± 0.119	0.255 ± 0.080	$\mathbf{0.471 \pm 0.082}$	0.407 ± 0.053
	IQ-Learn	-0.246 ± 0.138	0.117 ± 0.131	$\mathbf{0.332 \pm 0.035}$	0.393 ± 0.047
	ILEED	0.250 ± 0.034	$\mathbf{0.267 \pm 0.081}$	0.203 ± 0.060	0.005 ± 0.219

Table 1: WS of each IL algorithm over the demonstrator set, $\mathcal{I}$. Each algorithm was trained with distinct datasets filtered by various methods: (1) **Original**: utilizing the full dataset for training; (2) **RI**: filtering the dataset using the RI indicator; (3) **EL**: filtering the dataset using the EL indicator; and (4) **Best**: employing only the data generated by the dominant demonstrator, which serves as an oracle method. Bold highlights the best performance among Original, RI, and EL, while underline shows if the 'Best' method achieves the highest performance overall. Higher is better.

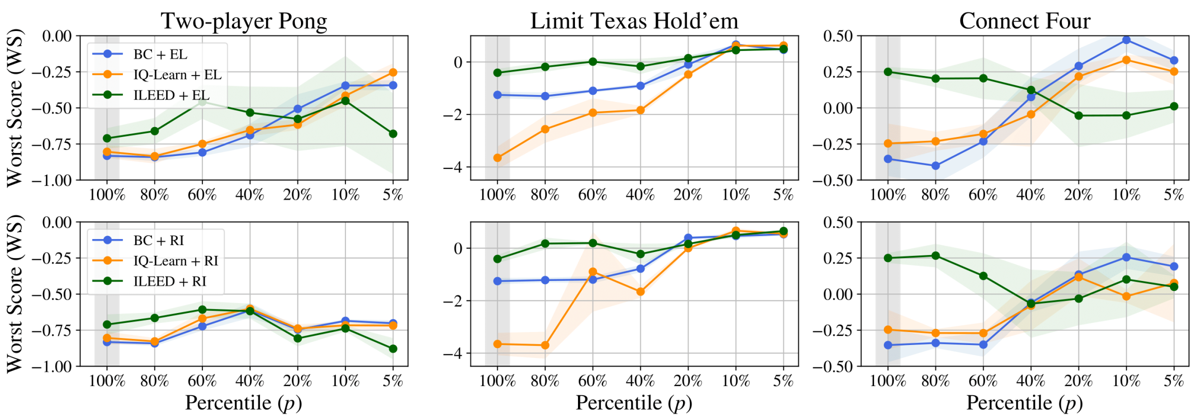


Figure 4: WS of each IL algorithm across different percentile (p) values for each indicator. The grey-shaded region represents the model trained on the original dataset, equivalent to the vanilla algorithm. Moving further to the right in the subfigure indicates a decrease in the data used. Higher is better.

cording to randomness is equivalent to reducing valid data, which results in worse performance.

Sensitivity analysis. Figure 4 shows the performance for each IL algorithm across different percentile values for each indicator. For the BC and IQ-Learn algorithms, the RI and EL methods provide improved performance in all the cases. Although ILEED is designed to learn from diverse demonstrators, the RI and EL methods can be effectively used in some environments because ILEED struggles to distinguish the dominant policy in a multi-agent environment. For the EL method, due to the significant decrease in the size of the filtered dataset, a drop in performance from $p = 0.1$ to $p = 0.05$ is commonly observed. In contrast, in the range of $p \geq 0.1$, the overall performance is enhanced as p decreases. EL is a reliable indicator since it has a few reward-labeled data as anchors, while RI solely takes estimated randomness as evaluation metrics. The result of the RI method across different p's shows less stable behavior, as the optimal results are achieved on $p = 0.4$ or $p = 0.1$ in different game scenarios. However, choosing a relatively small p for an unknown dataset is a preferred option since the most proficient demonstrators usually have the most stable strategies.

Conclusion

In this work, we proposed an effective framework, STRIL, to extract the representations of the offline trajectories and enhance imitation learning methods in multi-agent games. We designed a P-VRNN network, which shows extraordinary results in learning the strategy representations of trajectories without requiring player identification. We then defined two indicators, RI and EL, for imitation learning. We can estimate RI and EL by utilizing the strategy representation and subsequently filter the offline dataset with the indicators. The imitation learning algorithms show significant performance improvements with the filtered datasets.

In future work, we plan to utilize the P-VRNN as a customized behavior prediction model and explore the geometry of the strategy representation space. Additionally, we aim to develop IL methods that integrate the indicators beyond simply filtering the dataset.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grants 72293575, as well as by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. RS-2024-00410082)

References

Andrychowicz, O. M.; Baker, B.; Chociej, M.; Jozefowicz, R.; McGrew, B.; Pachocki, J.; Petron, A.; Plappert, M.; Powell, G.; Ray, A.; et al. 2020. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1): 3–20.

Bansal, T.; Pachocki, J.; Sidor, S.; Sutskever, I.; and Mordatch, I. 2018. Emergent Complexity via Multi-Agent Competition. In *International Conference on Learning Representations*.

Beliaev, M.; Shih, A.; Ermon, S.; Sadigh, D.; and Pedarsani, R. 2022. Imitation learning by estimating expertise of demonstrators. In *International Conference on Machine Learning*, 1732–1748. PMLR.

Berner, C.; Brockman, G.; Chan, B.; Cheung, V.; Debiak, P.; Dennison, C.; Farhi, D.; Fischer, Q.; Hashme, S.; Hesse, C.; et al. 2019. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*.

Bojarski, M.; Del Testa, D.; Dworakowski, D.; Firner, B.; Flepp, B.; Goyal, P.; Jackel, L. D.; Monfort, M.; Muller, U.; Zhang, J.; et al. 2016. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*.

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

Chen, J.; Yuan, B.; and Tomizuka, M. 2019. Model-free deep reinforcement learning for urban autonomous driving. In *2019 IEEE intelligent transportation systems conference (ITSC)*, 2765–2771. IEEE.

Chen, L.; Paleja, R.; and Gombolay, M. 2021. Learning from suboptimal demonstration via self-supervised reward regression. In *Conference on robot learning*, 1262–1277. PMLR.

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y. 2015. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28.

Ding, Y.; Florensa, C.; Abbeel, P.; and Phielipp, M. 2019. Goal-conditioned imitation learning. *Advances in neural information processing systems*, 32.

Fei, C.; Wang, B.; Zhuang, Y.; Zhang, Z.; Hao, J.; Zhang, H.; Ji, X.; and Liu, W. 2020. Triple-GAIL: a multi-modal imitation learning framework with generative adversarial nets. *arXiv preprint arXiv:2005.10622*.

Franzmeyer, T.; Elkind, E.; Torr, P.; Foerster, J. N.; and Henriques, J. F. 2024. Select to Perfect: Imitating desired behavior from large multi-agent data. In *The Twelfth International Conference on Learning Representations*.

Fujimoto, S.; Meger, D.; and Precup, D. 2019. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, 2052–2062. PMLR.

Garg, D.; Chakraborty, S.; Cundy, C.; Song, J.; and Ermon, S. 2021. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems*, 34: 4028–4039.

Grover, A.; Al-Shedivat, M.; Gupta, J.; Burda, Y.; and Edwards, H. 2018. Learning policy representations in multiagent systems. In *International conference on machine learning*, 1802–1811. PMLR.

Hausman, K.; Chebotar, Y.; Schaal, S.; Sukhatme, G.; and Lim, J. J. 2017. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. *Advances in neural information processing systems*, 30.

Heinrich, J.; and Silver, D. 2016. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*.

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

Kaelbling, L. P.; Littman, M. L.; and Cassandra, A. R. 1998. Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2): 99–134.

Kim, G.-H.; Seo, S.; Lee, J.; Jeon, W.; Hwang, H.; Yang, H.; and Kim, K.-E. 2022. Demodice: Offline imitation learning with supplementary imperfect demonstrations. In *International Conference on Learning Representations*.

Kumar, A.; Zhou, A.; Tucker, G.; and Levine, S. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191.

Littman, M. L. 1994. Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, 157–163. Elsevier.

Lynch, C.; Khansari, M.; Xiao, T.; Kumar, V.; Tompson, J.; Levine, S.; and Sermanet, P. 2020. Learning latent plans from play. In *Conference on robot learning*, 1113–1132. PMLR.

Mandlekar, A.; Booher, J.; Spero, M.; Tung, A.; Gupta, A.; Zhu, Y.; Garg, A.; Savarese, S.; and Fei-Fei, L. 2019. 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)*, 1048–1055. IEEE.

Mandlekar, A.; Xu, D.; Wong, J.; Nasiriany, S.; Wang, C.; Kulkarni, R.; Fei-Fei, L.; Savarese, S.; Zhu, Y.; and Martín-Martín, R. 2021. What Matters in Learning from Offline Human Demonstrations for Robot Manipulation. In *5th Annual Conference on Robot Learning*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Graves, A.; Antonoglou, I.; Wierstra, D.; and Riedmiller, M. 2013. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

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

Ross, S.; Gordon, G.; and Bagnell, D. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 627–635. JMLR Workshop and Conference Proceedings.

Sasaki, F.; and Yamashina, R. 2020. Behavioral cloning from noisy demonstrations. In *International Conference on Learning Representations*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sharma, P.; Mohan, L.; Pinto, L.; and Gupta, A. 2018. Multiple interactions made easy (mime): Large scale demonstrations data for imitation. In *Conference on robot learning*, 906–915. PMLR.

Terry, J.; Black, B.; Grammel, N.; Jayakumar, M.; Hari, A.; Sullivan, R.; Santos, L. S.; Dieffendahl, C.; Horsch, C.; Perez-Vicente, R.; et al. 2021. Pettingzoo: Gym for multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 15032–15043.

Vinyals, O.; Babuschkin, I.; Czarnecki, W. M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D. H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature*, 575(7782): 350–354.

Xu, H.; Zhan, X.; Yin, H.; and Qin, H. 2022. Discriminator-weighted offline imitation learning from suboptimal demonstrations. In *International Conference on Machine Learning*, 24725–24742. PMLR.

Yang, M.; Levine, S.; and Nachum, O. 2021. Trail: Near-optimal imitation learning with suboptimal data. *arXiv preprint arXiv:2110.14770*.

Zha, D.; Lai, K.-H.; Huang, S.; Cao, Y.; Reddy, K.; Vargas, J.; Nguyen, A.; Wei, R.; Guo, J.; and Hu, X. 2020. RLCard: A Platform for Reinforcement Learning in Card Games. In *IJCAI*.

Zhang, S.; Cao, Z.; Sadigh, D.; and Sui, Y. 2021. Confidence-aware imitation learning from demonstrations with varying optimality. *Advances in Neural Information Processing Systems*, 34: 12340–12350.