Collapsing Sequence-Level Data-Policy Coverage via Poisoning Attack in Offline Reinforcement Learning

Xue Zhou[1] **Dapeng Man**[1] **Chen Xu**[*1] **Fanyi Zeng**[1] **Tao Liu**[1] **Huan Wang**[1] **Shucheng He**[1] **Chaoyang Gao**[1] **Wu Yang**[1]

[1]College of Computer Science and Technology, Harbin Engineering University, China

Abstract

Offline reinforcement learning (RL) heavily relies on the coverage of pre-collected data over the target policy's distribution. Existing studies aim to improve data-policy coverage to mitigate distributional shifts, but overlook security risks from insufficient coverage, and the single-step analysis is not consistent with the multi-step decision-making nature of offline RL. To address this, we introduce the sequence-level concentrability coefficient to quantify coverage, and reveal its exponential amplification on the upper bound of estimation errors through theoretical analysis. Building on this, we propose the Collapsing Sequence-Level Data-Policy Coverage (CSDPC) poisoning attack. Considering the continuous nature of offline RL data, we convert state-action pairs into decision units, and extract representative decision patterns that capture multi-step behavior. We identify rare patterns likely to cause insufficient coverage, and poison them to reduce coverage and exacerbate distributional shifts. Experiments show that poisoning just 1% of the dataset can degrade agent performance by 90%. This finding provides new perspectives for analyzing and safeguarding the security of offline RL.

1 INTRODUCTION

Offline reinforcement learning (RL) leverages pre-collected static datasets for policy learning, avoiding the high costs and risks of online exploration [Levine et al., 2020]. This approach has demonstrated significant potential in domains such as robotic control [Chebotar et al., 2021] and autonomous driving [Diehl et al., 2023].

Despite offline RL's potential, its performance heavily depends on data-policy coverage [Munos, 2003], defined as how well the pre-collected data's empirical distribution matches the target policy's occupancy distribution [Agarwal et al., 2019]. Insufficient coverage exacerbates distributional shift [Kumar et al., 2019], degrading policy performance by increasing value estimation errors [Wang et al., 2023].

Current research focuses on single-step coverage to improve algorithms, ensuring the target policy more effectively relies on available data distributions to mitigate distribution shift [Fujimoto et al., 2018, Kidambi et al., 2020b]. However, they overlook the multi-step decision-making nature of RL, where policies depend on sequences of actions rather than individual steps [Sutton and Barto, 2018]. Recent studies propose sequence-level methods [Bar-David et al., 2023, Saanum et al., 2023], but the impact of insufficient sequence-level coverage on policy learning remains underexplored.

Building on existing research that uses the concentrability coefficient to quantify single-step coverage, we extend this concept to the sequence-level. Through theoretical analysis, we reveal that insufficient sequence-level coverage exponentially increases the upper bound of estimation errors. This issue also exposes potential vulnerabilities in offline RL to adversarial attacks, attackers could exploit data poisoning to manipulate coverage and degrade agent performance.

Following this idea, we design a Collapsing Sequence-Level Data-Policy Coverage (CSDPC) attack in offline RL, employing data poisoning to reduce sequence-level coverage. Since offline RL data is predominantly continuous, making direct coverage calculation infeasible, we first cluster the single-step state-action pairs, converting similar behaviors into unified decision units. Next, we extract representative decision patterns, defined as consecutive decision units with repetitions removed, to capture essential behavioral logic. We identify rare patterns that occur infrequently and are most likely to cause coverage insufficiency. Finally, we minimally perturb the data to eliminate these rare patterns, significantly reducing learnable patterns and amplifying distribution shifts. Experimental results validate the effectiveness of our approach. Our contributions are as follows:

Proceedings of the 41st Conference on Uncertainty in Artificial Intelligence (UAI 2025), PMLR 244:5084–5098.

- We analyze distributional shifts in offline RL from the perspective of data-policy coverage. Considering RL's multi-step decision-making nature, we use theory and targeted poisoning to demonstrate the critical impact of sequence-level coverage insufficiency.
- We introduce the sequence-level concentrability coefficient to quantify coverage, and show that insufficient coverage exponentially amplifies learning errors.
- We develop a poisoning attack to collapse data-policy coverage by precisely eliminating rare decision patterns, demonstrating that minimal perturbations can amplify distributional shifts and disrupt policy learning.
- Experiments on multiple offline RL environments show that just 5% perturbation magnitude on 1% data can reduce agent performance by 90%, and can evade existing detection methods. Moreover, even with only 1% data access, the attack achieves 86% of the effectiveness compared to full data access.

Our study reveals potential data quality issues and security vulnerabilities in offline RL under sequence-level distributional shifts, offering new directions for future research and protective measures.

2 RELATED WORK

Offline RL Due to the limitations of online RL in resources and safety [Levine et al., 2016, Silver et al., 2017], an increasing body of research is shifting towards offline RL, which learns policies from pre-collected datasets [Levine et al., 2020, Kumar et al., 2020c, Fujimoto and Gu, 2021].

The offline RL is formalized within a Markov decision process, defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$, where $\mathcal{S}$ denotes the state space, $\mathcal{A}$ denotes the action space, $T(s'|s,a)$ denotes the transition distribution, $R(s,a)$ denotes the reward function, and $\gamma \in (0,1)$ denotes the discount factor. The goal of offline RL is to find a policy $\pi(a|s)$ that maximizes the expected cumulative discounted rewards from a pre-collected dataset D, formalized as $\pi = arg \max_{\pi} \mathbb{E}_{\tau \sim D}[\sum_{t=0}^{L} \gamma^t R(s,a)]$, $\tau_t = (s_t, a_t, r_t, s_{t+1})$ is a trajectory at time t, L is the trajectory's maximum length, and $\mathbb{E}$ denotes expectation [Sutton and Barto, 2018].

Data-Policy Coverage A pivotal challenge in offline RL lies in learning when the behavioral policy's empirical distribution fails to adequately cover the target policy's occupancy distribution [Prudencio et al., 2022]. Existing research leverages coverage metrics to address distributional shifts: certain studies incorporate behavioral regularization to reduce out-of-distribution actions [Wu et al., 2019], while others use pessimistic penalties to constrain the policy to well-covered regions[Yu et al., 2020, Kidambi et al., 2020a]. However, the analysis of sequence-level coverage deficiencies in datasets and their impact on learning outcomes is still lacking. To better understand its effect on offline RL performance, this paper conducts a systematic study from both theoretical and empirical perspectives.

Poisoning Attack Given that malicious attackers can manipulate decisions through erroneous data, exposing offline RL's sensitivity to such risks and exploring mitigation measures are crucial for developing robust algorithms and gaining user trust [Rangi et al., 2022, Gong et al., 2024]. Poisoning attack serves as a potent tool for evaluating algorithm vulnerabilities, and numerous studies in the online RL field have leveraged them to assist developers in gaining insights into system vulnerabilities [Zhang et al., 2020a,b, 2021, Sun et al., 2021, Qu et al., 2021, Sun et al., 2022, Standen et al., 2023]. Moreover, some studies have employed various methods ranging from heuristic to model-driven approaches to help researchers identify weak points in RL systems [Lin et al., 2017, Kos and Song, 2017, Sun et al., 2020, Yu et al., 2023]. However, these methods typically require real-time access to online training parameters or other observations, which are inapplicable in offline environments lacking interaction and synchronous updates.

As offline RL has garnered increased attention, its security challenges have come into focus. Early studies attacked batch RL, but the high cost of attacks at every step posed challenges [Ma et al., 2019]. Subsequently, researchers attempted to formulate attacks as optimization problems, devising strategies based on different costs [Rakhsha et al., 2020]. There are theoretical analyses of the security of offline RL [Rangi et al., 2022]. Additionally, some attacks introduce triggers in datasets, conditional on the attacker's control over both training and testing stages, thus imposing stringent preconditions [Gong et al., 2024].

Previous studies assume attackers have elevated privileges, and neglect the impact of sequential time steps. We propose an attack strategy to investigate the effects of sequence-level coverage insufficiency.

3 PROBLEM FORMULATION

3.1 THREAT MODEL

Following the setup in existing work [Ma et al., 2019, Rakhsha et al., 2020, Gong et al., 2024], the application idea of offline RL is that anyone can serve as a data provider to openly share their experience data, and developers can use open-source data to train RL agents to reduce consumption.

Attacker's Privileges Set the attacker as a malicious data provider or processor, capable of poisoning parts of the offline dataset. To ensure stealth, the poisoning rate and perturbation magnitude must be minimized to avoid detection.

Considering a more realistic scenario, we discuss the threats posed by attackers with limited data access.

Attacker's Goal The attacker's objective is to manipulate the agent by poisoning the dataset, inducing it to learn a policy that minimizes cumulative rewards or even incurs penalties, ultimately leading to erroneous actions based on the learned poisoned policy.

Attacker's Knowledge Attackers require no domain knowledge (e.g., dynamics and perception in robotics, computer vision, or vehicle dynamics in autonomous driving). Before the attack occurs, we assume the attacker can train the agent and make an approximate estimate, a method widely used in previous work [Zhang et al., 2020b].

3.2 IMPACT OF SEQUENCE-LEVEL COVERAGE

The performance of offline RL policy learning heavily relies on the empirical data distribution's coverage of the regions accessed by the target policy [Levine et al., 2020]. Insufficient coverage can lead to significant errors in the target policy due to distributional shifts, thereby degrading policy performance. Researchers quantified this by introducing the concentratability coefficient C [Munos, 2003]:

$$\begin{aligned} C &= \sup_{(s,a)\in\mathcal{I}_\mu} \frac{d^\pi(s,a)}{\mu(s,a)}, \\ d^\pi(s,a) &= (1-\gamma)\sum_{t=0}^{\infty}\gamma^t \Pr(s_t = s, a_t = a \mid \pi), \end{aligned} \tag{1}$$

where $d^\pi(s,a)$ is the occupancy distribution under the target policy π, once π and the environment's transition probabilities P are fixed, $d^\pi(s,a)$ becomes a constant measure describing how frequently (s,a) is visited by π. The probability $\Pr$ depends solely on P and π. $\mu(s,a)$ denotes the empirical distribution of state-action pairs in the data collected by the behavior policy μ. $\mathcal{I}_\mu = \{(s,a) \mid \mu(s,a) > 0\}$ ensures that $\mu(s,a) > 0$ to prevent C from diverging. This ratio quantifies the disparity between the π visitation frequency at (s,a) and its coverage in the dataset. sup represents taking the supremum over all possible (s,a) pairs, it measures the worst-case distributional shift between the target policy and the empirical distribution. Due to the limited coverage of offline datasets, if there exist any (s,a) pairs such that $d^\pi(s,a) \geq \mu(s,a)$, this results in $C \geq 1$ [Chen and Jiang, 2019]. Moreover, when the dataset contains very sparse (s,a) pairs ($\mu(s,a)$ is small) but $d^\pi(s,a)$ remains significant, C can significantly increase. This reflects insufficient coverage, which exacerbates distributional shifts, leading to inadequate policy learning and a decline in model performance [Lee et al., 2020].

Researchers have evaluated the impact of an increased C on the offline RL learning process using action-value function as an example, based on the analysis of the concentrability coefficient and error propagation from references [Munos, 2003, Chen and Jiang, 2019], the upper bound of the error between the action-value function of the target policy Q^π and the optimal Q^{π^*} can be derived as:

$$\mathbb{E}_{(s,a)\sim d^\pi}\left[Q^{\pi^*}(s,a) - Q^\pi(s,a)\right] \leq \frac{2R_{\max}C}{1-\gamma}\epsilon, \tag{2}$$

where $R_{\max}$ is the maximum possible value of the reward function, and ϵ denotes the internal approximation (or regression) error under the data distribution μ (e.g., due to limited samples or function approximation), and does not directly depend on C. The equation indicates that when the data coverage is insufficient, the concentrability coefficient C then serves as a shift-amplifier, mapping this internal error from μ to the target policy distribution d^π, ultimately enlarging the Q function estimation bias.

Although the concentrability coefficient is instrumental in enhancing policy learning [Wang et al., 2023], existing studies remain constrained to measuring deviations at individual state-action pairs, overlooking the multi-time-step decision-making nature of reinforcement learning. Recent studies have increasingly focused on sequence-level analysis [Bar-David et al., 2023, Saanum et al., 2023]. To better characterize the distributional properties of offline RL data, we extend the concentrability coefficient to the sequence level. Sequence-level concentrability coefficient C_τ is defined as:

$$C_\tau = \sup_{\tau\in\mathbb{T}_\mu} \frac{d^\pi(\tau)}{\mu(\tau)}, \tag{3}$$

where $\tau = (s_0, a_0), (s_1, a_1), \ldots, (s_{l-1}, a_{l-1})$ is a sequence of l time steps, $\mu(\tau)$ denotes the probability distribution of this sequence in the offline data, and $\mathbb{T}_\mu = \{\tau \mid \mu(\tau) > 0\}$ ensures $\mu(\tau) > 0$ to prevent the ratio remains finite, and $d^\pi(\tau)$ represents the probability distribution of generating this sequence under the target policy. By decomposing the trajectory probabilities, we obtain:

$$\begin{aligned} \frac{d^\pi(\tau)}{\mu(\tau)} &= \frac{\prod_{t=0}^{l-1}\left[\pi(a_t \mid s_t)P(s_{t+1} \mid s_t, a_t)\right]}{\prod_{t=0}^{l-1}\left[\mu(a_t \mid s_t)P(s_{t+1} \mid s_t, a_t)\right]} \\ &= \prod_{t=0}^{l-1}\frac{\pi(a_t \mid s_t)}{\mu(a_t \mid s_t)}. \end{aligned} \tag{4}$$

According to Equation 1, C is the supremum, implying that for each time step $\frac{\pi(a_t|s_t)}{\mu(a_t|s_t)} \leq C$, and assuming the transition probabilities P are consistent under both the behavior and target policies [Sutton and Barto, 2018], then:

$$C_\tau = \sup_{\tau\in\mathcal{T}_\mu} \frac{d^\pi(\tau)}{\mu(\tau)} = \sup_{\tau\in\mathcal{T}_\mu} \prod_{t=0}^{l-1}\frac{\pi(a_t \mid s_t)}{\mu(a_t \mid s_t)} \leq \prod_{t=0}^{l-1} C = C^l. \tag{5}$$

This suggests that with insufficient coverage of some sequence, the sequence-level concentrability coefficient C_τ

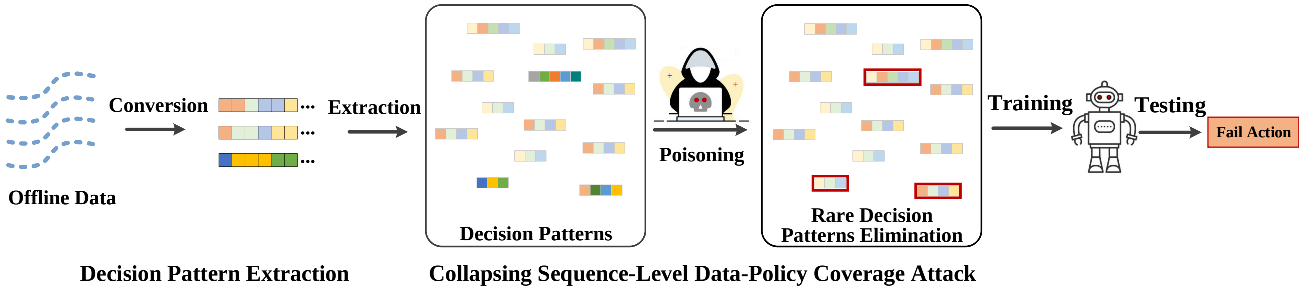

Figure 1: The workflow of the CSDPC. We first convert continuous data into decision units and merge consecutively repeated units to extract representative decision patterns. Next, we poison the dataset to transform rare patterns into common ones, thereby reducing coverage and exacerbating distributional shifts, leading to a decline in the performance of poisoned agents.

can reach up to the l power of the single-step coefficient C ($C \geq 1$). We further quantify Q-value estimation errors at the sequence level. Given that value estimation in RL incorporates discount weights over time steps to reflect the decay of future rewards or errors [Sutton and Barto, 2018], we adopt a discounted weighted sum when measuring sequence-level Q-value errors. Based on Equation 2, the weighted Q-value estimation error at the sequence level is:

$$\begin{aligned}&\mathbb{E}_{\tau\sim d^{\pi}}\left[\sum_{t=0}^{l-1}\gamma^{t}\left(Q^{\pi^*}(s_t,a_t)-Q^{\pi}(s_t,a_t)\right)\right]\\&\leq\frac{2R_{\max}C_{\tau}}{1-\gamma}\epsilon\leq\frac{2R_{\max}C^{l}}{1-\gamma}\epsilon.\end{aligned} \tag{6}$$

This shows that when certain sequences occur infrequently ($\mu(\tau)$ is small), the increase in C_τ indicates insufficient coverage, leading to exponential growth in value estimation error bounds and significantly impacting the performance of the target policy. Altering the distribution of these rare sequences can exacerbate distributional shifts, further amplifying estimation errors and degrading policy performance, which offers a new perspective on data poisoning.

4 METHOD

The above analysis indicates that insufficient sequence-level coverage exponentially amplifies the upper bound on the estimation error of the Q function. To further verify the impacts, we design the Collapsing Sequence-Level Data-Policy Coverage (CSDPC) poisoning attack to reduce sequence-level coverage.

In realistic offline RL scenarios, such as robotics and autonomous driving [Fu et al., 2020], high-dimensional continuous data make it infeasible to compute the sequence-level coverage directly, so we first converse state-action pairs into decision units to enable coverage analysis (Section 4.1). Next, we extract decision patterns from sequences by removing repetitions, compactly representing multi-step behavioral logic (Section 4.2). We then identify rare decision patterns that correspond to low-coverage regions and are most likely to amplify distributional shifts and estimation errors (Section 4.3). Finally, we inject minimal perturbations to eliminate these patterns, reducing sequence-level coverage (Section 4.4). Figure 1 is the CSDPC attack workflow, and detailed steps in Appendix A.

4.1 DECISION UNIT CONVERSION

To address the challenge of handling continuous data in offline RL, a natural idea is to discretize continuous state-action pairs into countable units. The goal is to map each $(s, a) \in \mathcal{S} \times \mathcal{A}$ to a discrete unit $u_j \in U$, where $U = u_1, u_2, \ldots, u_k$ is a finite set of decision units. This mapping can be formalized as:

$$\phi(s_t, a_t) = \text{Discretize}(s_t, a_t), \tag{7}$$

where $\phi : \mathcal{S} \times \mathcal{A} \to U$ is the discretization function. We employ the classical k-means clustering algorithm to achieve this discretization, and the optimal number of clusters k is determined using the elbow method [Jain, 2008], each cluster is represented by a unit u_j.

However, previous studies have emphasized that high-dimensional raw inputs usually contain redundant or noisy information [Mnih et al., 2015]. Therefore, direct clustering of raw (s, a) may not be effective in capturing potential decision-making behaviors. Some research suggests that higher-level feature representations derived from raw data better capture the essential decision-making logic of RL agents [Fujimoto et al., 2018, Kumar et al., 2020b].

Inspired by this, we train an RL agent on a clean dataset D, and use its trained encoder network to extract meaningful features from the raw data. Specifically, for each time step, we encode the state-action pair (s, a) into a feature vector f_t. Formally, Equation 7 can be formalized as:

$$\phi(f_t) = \text{K-means}[\text{Encoder}(s, a)]. \tag{8}$$

The encoder maps the raw state-action pair to a higher-level, decision-relevant feature representation. Compared to clustering raw data, this method provides a more precise and agent-aware approach to decision unit conversion.

4.2 DECISION PATTERN EXTRACTION

After converting state-action pairs into decision units, the next step is to extract sequence-level decisions from sequences to compute coverage.

Given a trajectory of length L, from which we extract a sequence $\tau_t = (s_t, a_t), (s_{t+1}, a_{t+1}), \dots, (s_{t+l}, a_{t+l})$ with a sequence length l satisfying $2 \leq l \leq L$, where t denotes the starting index. After discretizing the state-action pairs into decision units, each (s, a) is mapped to a clustering label $u \in \{1, 2, \dots, k\}$, resulting a sequence of behaviors represented by the cluster labels:

$$\tau_t = u_t, u_{t+1}, \dots, u_{t+l}. \tag{9}$$

The decision space for sequences of length l is too large by k^l, thus complicating the analysis. Meanwhile, consecutive decision units often represent similar behaviors. We merge consecutively repeated units to obtain the de-duplicated consecutive decision units, defined as the decision pattern p.

This step ensures the representation highlights meaningful behavioral changes while reducing redundant repetition. This deduplication strategy is particularly suitable for continuous or near-continuous control tasks (e.g., robotics, autonomous driving), where repetitive actions dominate most time steps. For instance, straight-line driving occurs frequently, potentially overshadowing sparse yet crucial behaviors such as turning. Deduplication helps compress redundant behaviors and better reflect the structure of meaningful decision patterns.

4.3 RARE DECISION PATTERN IDENTIFICATION

Theoretical analysis indicates that a higher sequence-level concentrability coefficient C_τ corresponds to lower data-policy coverage, thereby exacerbating distributional shifts. To exploit this vulnerability, our attack aims to increase C_τ, thereby effectively reducing coverage and amplifying the impact of distributional shifts. The most effective approach is to decrease the frequency of rare sequences, as these sequences are most likely the contributors to low-coverage regions. Thus, it is necessary to identify rare sequences. We previously extracted representative decision patterns from sequences. Next, we identify rare decision patterns.

To identify rare decision patterns, we first extract decision patterns p from all sequences in the dataset, and compute the occurrence frequency of each pattern denoted as $O(p)$. Next, we rank all patterns by their occurrence frequency $O(p)$. Based on a limited poisoning rate ρ $(0 < \rho < 1)$, selecting the patterns with the lowest frequencies to form the rare decision pattern set $\mathcal{P}$.

These rare patterns correspond to the sequence-level sparse regions identified in the theoretical analysis and most likely lead to an increase in C_τ. The rare decision pattern set $\mathcal{P}$ serves as the target for subsequent poisoning operations.

4.4 COVERAGE REDUCTION VIA POISONING

To achieve the object of reducing sequence-level data-policy coverage, the attack must ensure both effectiveness and stealthiness, emphasizing imperceptibility to simulate real-world adversarial scenarios.

Specifically, we devise a perturbation strategy based on two crucial components: First, introducing a stealthiness constraint for the perturbation magnitude to ensure the attack proceeds without causing conspicuous anomalies to evade detection; Second, we reduce coverage by erasing rare decision patterns from the dataset through poisoned perturbations, thereby increasing C_τ.

Stealth Constraint Imposition Our objective is to explore a more stealth setup that minimizes the attack's magnitude and adapts it to the size of the original data. We introduce a perturbation ζ_t, generating the poisoned state-action pair $(s_t + \zeta_t^s, a_t + \zeta_t^a)$. The perturbation ζ_t is subject to the following constraints:

$$\parallel \zeta_t^s \parallel_\infty < \eta \cdot \parallel s_t \parallel_\infty \ , \ \parallel \zeta_t^a \parallel_\infty < \eta \cdot \parallel a_t \parallel_\infty, \tag{10}$$

where η represents the perturbation ratio, almost set to 0.05 in our paper. This ensures that the perturbations remain a small portion of the original values, thus maintaining the stealth of the attack.

Rare Decision Pattern Elimination To effectively reduce sequence-level coverage, our attack focuses on transforming the rarest decision patterns into the most frequent ones through targeted poisoned perturbations. By increasing the occurrence of these rare patterns, thereby significantly minimize sequence-level coverage.

For each raw sequence τ corresponding to a rare decision pattern $p \in \mathcal{P}$, we generate multiple potential poisoned sequences $\tau^1, \tau^2, \dots, \tau^n$ under a stealth constraint. These multiple potential poisoned sequences ensures that the perturbation most capable of replacing the raw rare pattern can be identified, thereby maximizing the attack's impact while adhering to the stealth constraint. Each perturbed sequence undergoes clustering and extraction operations to obtain corresponding decision patterns $p^1, p^2, \dots, p^n$. We then compute the occurrence counts O_{p^i} for each poisoned decision pattern in the dataset.

To maximize the attack's impact, we select the perturbed

Table 1: The ACR obtained by the agent in various environments, the values in parentheses are the AER.

Algorithms	Walker2D			Hopper			Half			Carla		
	Clean	Raw	Feature	Clean	Raw	Feature	Clean	Raw	Feature	Clean	Raw	Feature
CQL	3132	438(86%)	263(92%)	3158	380(88%)	234(93%)	4822	626(87%)	513(89%)	191	61(68%)	30(84%)
BEAR	2593	221(91%)	172(93%)	2119	215(90%)	131(94%)	4290	516(88%)	421(90%)	89	26(71%)	11(87%)
BCQ	2341	365(84%)	223(90%)	2823	280(89%)	203(93%)	4694	904(81%)	772(84%)	466	153(67%)	72(85%)
BC	744	107(86%)	62(92%)	3450	384(89%)	226(93%)	4017	516(87%)	400(90%)	384	128(67%)	70(82%)
Average	2203	285(87%)	161(92%)	2613	315(89%)	199(93%)	4456	641(86%)	527(88%)	283	92(68%)	46(85%)
	Raw Data AER 83%						Feature Data AER 90%					

sequence τ^i that results in the highest occurrence count O_{p^i}, as it most effectively replaces the raw rare decision pattern. This selected poisoned sequence τ^n replaces the raw sequence τ. Applying this poisoning process to all data results in the poisoned dataset D'.

This replacement operation significantly reduces the presence of rare decision patterns in the dataset, thereby lowering sequence-level coverage. As a result, the target policy is forced to rely on a limited set of decision patterns, exacerbating distributional shifts.

5 EXPERIMENTS AND ANALYSIS

In this section, we conduct a comprehensive ablation study on the Collapsing Decision Pattern Diversity (CSDPC) attack. We evaluate these attacks under various offline RL settings across diverse environments and provide an in-depth analysis of the experimental results.

5.1 EXPERIMENTAL SETTINGS

Settings Considering stealthiness, we set our poisoning proportion range to $\rho = \{1\%, 5\%, 10\%, 20\%\}$, with most experiments were conducted at $\rho = 1\%$ and $\rho = 5\%$. The perturbation magnitude ranged as $\eta = \{0.05, 0.15, 0.25\}$, where most of the experiments were performed with constraints of $\eta = 0.05$. Unless specified otherwise, the attacks mentioned in the experimental section are based on the raw data. The detailed experimental settings is outlined in Appendix B.

Evaluation Metrics The Average Cumulative Reward (ACR), defined as $R = \frac{1}{T}\sum_{t=0}^{T} r_t$, directly measures the agent's task performance in the test environment. In our experiments, T is set to 50, meaning the ACR reflects the agent's average performance across 50 trajectories in the test environment.

To evaluate the impact of different attack methods on agent performance, we adopt the commonly used metric: attack effectiveness rate (AER), which quantifies the performance degradation of the poisoned agent compared to the clean agent. AER is calculated as:

$$\text{AER} = \frac{R_{clean} - R'_{poisoned}}{R_{clean}} * 100\%, \tag{11}$$

where $R'_{poisoned}$ is the average cumulative reward of the poisoned model in the trigger environment. A higher AER indicates greater attack effectiveness.

5.2 OVERALL RESULT OF CSDPC

Effectiveness Table 1 shows the attack outcomes in four tasks with a mere 1% poisoning rate, where CSDPC attack can reduce the agents' average cumulative rewards (ACR) by 83% when partitioning the decision space using raw data, as effective as the 10% poisoning rate in BAFFLE [Gong et al., 2024]. Moreover, when attackers can extract advanced features from the data, the attacks can diminish the agents' average performance by 90%. The $\rho = 5\%$ and more comparative results are provided in Appendix C.

These results show that our attack can significantly disrupt offline RL algorithms across various tasks with minimal cost. The effectiveness of the attack increases with the attacker's knowledge, indicating that the attacker can capture more crucial information during the learning process, thereby hindering the agent from learning optimal policies.

Generalizability Although our analysis is based on Q-values, the results in Table 1 reveal that the CSDPC attack remains effective against the BC algorithm, which does not use Q-values updates. Additionally, CSDPC attacks significantly degrade agent performance in complex tasks across different data types (e.g., numeric inputs for Mujoco robot tasks, and image inputs for Carla autonomous driving tasks). We further conduct additional experiments on AWAC Kumar et al. [2020a], a policy gradient method. Even $\rho = 1\%$ and encoder mismatch, the AER is 85% in Walker2D. These results indicate that reducing the coverage of rare decision patterns has a significant impact on various offline RL algorithms for a wide range of tasks. This is attributed to the fact that our attack method does not rely on any specific offline RL algorithm or task structure design, but rather impacts agents by targeting the essence of coverage.

Table 2: CSDPC attacks based on raw data were executed on the CQL agent in the Walker2D task under varying data privileges and perturbation magnitudes. Small-range data was randomly selected as consecutive time steps from the dataset.

Clean	η	$\rho = 1\%$					$\rho = 5\%$				
		Whole Data	20% Data	10% Data	5% Data	1% Data	Whole Data	20% Data	10% Data	5% Data	1% Data
3132	0.05	438(86%)	604(81%)	665(79%)	725(77%)	952(70%)	322(90%)	489(84%)	541(83%)	637(80%)	856(73%)
	0.15	361(88%)	530(83%)	564(82%)	680(78%)	905(71%)	213(93%)	438(86%)	478(85%)	542(83%)	737(76%)
	0.25	299(90%)	433(86%)	527(83%)	620(81%)	723(77%)	184(94%)	384(88%)	402(87%)	499(84%)	622(80%)

Table 3: Anomaly value-based detection results, $\rho = 5\%$.

Environments	Precision		Recall		F1-score	
	clean	poison	clean	poison	clean	poison
Hopper	8.0%	7.6%	24.3%	24.5%	11.6%	12.8%
Half	7.9%	9.0%	24.1%	25.1%	13.5%	12.7%
Walker2D	6.9%	6.9%	25.8%	24.8%	10.9%	11.9%
Average	7.6%	7.8%	24.7%	24.8%	12.0%	12.5%

Stealthiness To enhance the stealthiness of the attack, we explore the effectiveness of attacks when the attacker has only limited data upload or processing privileges. As demonstrated in Table 2, even with access to as little as 1% of the data, the attacker can significantly degrade the agent's performance (e.g., reducing returns by 70% under $\eta = 0.05, \rho = 1\%$). Moreover, the attack's effectiveness consistently increases as the attacker gains access to a larger privileges of the data. This trend suggests that having access to more data allows the attacker to identify and perturb a greater number of critical decision points within the agent's learning process, leading to more substantial damage.

We also investigate the impact of varying perturbation magnitudes on the attack effectiveness. We observe that larger perturbations can generate poisonous samples that reside in a different cluster from the raw data, eradicating more rare decision patterns, thus significantly reducing the rare decision pattern coverage in the dataset, as reflected in the results in Table 2. Another clear trend is that larger perturbation sizes can offset the diminished effectiveness of attacks due to restricted data access. For example, when the perturbation size is increased from 0.5 to 0.25, with the same poisoning rate, 5% data access can achieve the same attack effectiveness as with 20% data access.

To further evaluate the stealthiness of the CSDPC attack, we employed GradCon [Kwon et al., 2020], a widely used anomaly detection approach, to detect clean and poisoned data points. We generated poisoned data by injecting poison into the Walker2D dataset using the CQL algorithm with the CSDPC attack, $\rho = 5\%$. The results in Table 3 indicated no significant anomalies in the detection of poisoned data compared to clean data. Notably, the average F1 score for poisoning was only 12.5% using our method, in contrast to GradCon's over 87% F1 score on other datasets. This indicates that the anomalies were almost undetectable, thereby demonstrating the stealthiness of our approach.

Table 4: Perturbing consecutive and discrete time steps within the same sequence of CQL agents trained on the Walker2D dataset under full data privilege.

Data	$\rho = 1\%$		$\rho = 5\%$	
	Sequence	Discrete Steps	Sequence	Discrete Steps
Raw	478(85%)	539(83%)	372(88%)	443(86%)
Feature	377(88%)	419(87%)	243(92%)	348(89%)

Table 5: Attack results under different n-step CQL configurations in Walker2D.

Algorithm	ρ	n-step	Clean	ACR(AER)
CQL	1%	1	3132	438(86%)
		5	3296	297(91%)

5.3 ABLATION STUDY

Continuous vs. Discrete Poisoning To investigate whether continuous time steps have a greater impact on the learning process compared to discrete ones, we identify critical sequences of length 10 based on evaluation criteria. We perturb a random sequence of length 5 and discrete time steps spanning 5 intervals within each critical sequence using the CSDPC attack. The results in Table 4 indicate that within the same range, attack sequences lead to a greater degradation in the agent's performance, thus demonstrating a more significant impact on the learning process in continuous timesteps compared to discrete ones.

In addition, to assess the impact of multi-step corruption, we evaluate our attack on CQL with n-step = 5, which explicitly incorporates multi-step return estimation. As shown in Table 5, the agent experiences even greater performance degradation than the single-step version under the same poisoning budget, supporting our theoretical insight that sequence-level coverage insufficiency has a more pronounced effect in multi-step algorithms. These findings confirm that the more an algorithm relies on sequence information, the more vulnerable it becomes to sequence-level poisoning, further validating the practical relevance of our attack.

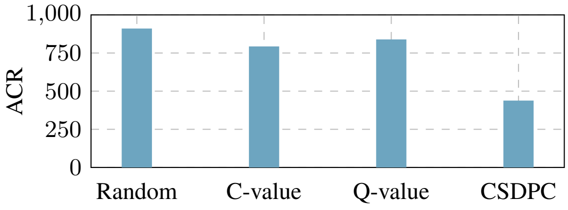

Figure 2: Identify critical sequences methods.

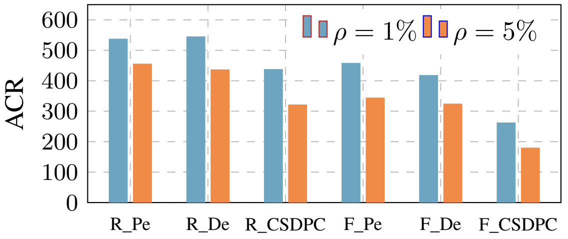

Figure 3: Different methods of poisoning.

Identifying Sequences To discuss the impact of sequences identified by different evaluation criteria on the attack, we utilize random selection, C-value [Lin et al., 2017], Q-value [Kos and Song, 2017], and our CSDPC attack method to determine critical sequences of length 5. Utilizing the perturbation method under constraints, we introduce perturbations with magnitudes within 0.05 to state-action pairs of the selected critical sequences. Figure 2 demonstrates the effectiveness of attacking different sequences. Results indicate that our method can accurately select sequences with a greater impact on the learning process.

This is may because high-value sequences often correspond to prevalent decision patterns (e.g., fast straight-line actions in autonomous driving tasks). As these patterns occur frequently in the dataset, the agent's learning of them is more stable, resulting in a lesser impact on the agent's performance when these patterns are perturbed. Conversely, CSDPC attack results in the absence of rare decision patterns within the dataset, considerably impairing the agent's capability in scenarios of inadequate learning, and subsequently precipitating a significant drop in performance.

Poisoning Methods To explore the effect of eliminating rare decision patterns on attacks, we compare the CSDPC attack method proposed in this paper with an approach that merely introduces perturbations under certain constraints (denoted as "Pe"), and those that directly delete sequences corresponding to rare decision patterns (denoted as "De"). Results depicted in Figure 3 indicate that the CSDPC attack is more effective under various poisoning rates and levels of attacker knowledge (where "R" represents the use of raw data, and "F" indicates the use of advanced features). Compared to these two methods, the CSDPC attack not only eliminates rare decision patterns but also alters the distribution of other common decision patterns, thereby exerting a greater negative impact on the agent.

Table 6: Impact of deduplication on attack performance, $\rho = 1\%$.

Setting	Clean	ACR	AER
w/o Deduplication	3132	971	69%
w/ Deduplication		438	86%

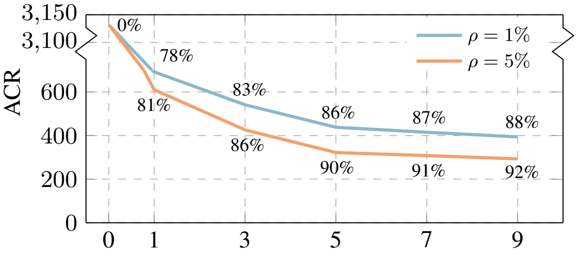

Figure 4: Different length of sequence.

Effect of Deduplication To evaluate the impact of our deduplication strategy, we conducted additional experiments on the Walker2D dataset. Compared to the original sequences, deduplication reduced the number of distinct decision patterns by nearly 80%, highlighting its role in removing repetition and preserving meaningful behavior changes. We further evaluated the impact on attack effectiveness using the CQL algorithm under a 1% poisoning rate with raw data. As shown in Table 6, deduplication significantly improves the attack's effectiveness.

Sequence Length To examine the influence of sequence length on attack effectiveness, we consider sequences of 1, 3, 5, 7, and 9 consecutive time steps. Our attack manifests in two variants for different sequence lengths: at length 1, it resembles a targeted method for single-time steps; at greater lengths, it serves as a perturbation method for the entire data. For stealth considerations, we avoid longer sequences, and unless otherwise stated, set the sequence length to 5 in all other experiments. The results in Figure 4 show that longer sequences lead to better attack outcomes. This is attributed to longer sequences that can capture more complex decision patterns, which are likely to be more prominent and critical in the dataset. In contrast, the decision patterns captured by shorter sequences become increasingly simple and transient. These brief sequences might only encompass a fraction of the agent's decision, resulting in a relatively minor impact on the agent's performance when attacked.

Poisoning Rate To delve deeper into the impact of poisoning rates on attack effectiveness, we conduct the CSDPC attack with varying poisoning rates against the CQL agent in the Walker2D environment. As shown in Figure 5, the trend

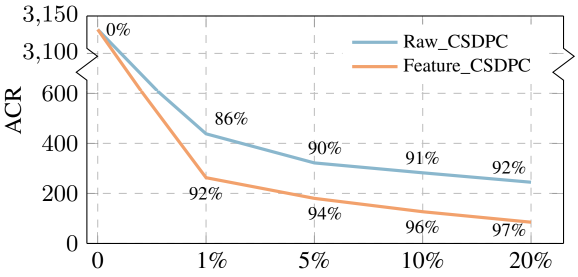

Figure 5: Different poisoning rate.

Table 7: Effect of cluster number k, $\rho = 1\%$.

Environment	Algorithm	Clean	k	ACR(AER)
Walker2D	CQL	3132	6	689(78%)
			8	438(86%)
			10	626(80%)

of average cumulative rewards varying with poisoning rates is depicted, $\eta = 0.05$. A clear trend emerges from the graph: as the poisoning rate ρ increases, the average cumulative reward obtained by the agent under CSDPC attacks significantly decreases. This result aligns with our expectation, as a higher poisoning ratio signifies the disappearance of more rare decision patterns in the dataset, and the data-policy coverage is significantly reduced. Consequently, this leads to greater degradation in the agent's performance.

Effect of cluster number k To evaluate the impact of the number of clusters used for discretizing state-action pairs, we conducted an ablation study with raw data. We compare three different settings for the number of clusters: $k = 6$, $k = 8$ (selected using the elbow method), and $k = 10$. The results in the Table 7 show that selecting k via the elbow method leads to the most effective attack performance, likely due to a better trade-off between merging similar behaviors and preserving discriminative structure.

6 DISCUSSION

In this paper, during the process of discretizing the state-action space, we prioritize computational efficiency for the attack, opting not to utilize a more refined method. This choice may have constrained the attack's effectiveness. In the future, we aim to develop more effective methods.

Our theoretical analysis based on the sequence-level concentrability coefficient assumes independence across steps, thus it does not explicitly account for correlated perturbations. Such temporal correlations may amplify coverage deficiencies beyond the provided theoretical bounds. Future work will explore modeling correlated biases across time steps to achieve tighter theoretical guarantees.

While this paper focuses on identifying and analyzing vulnerabilities, we acknowledge that robust defense mechanisms against sequence-level poisoning in offline RL remain largely unexplored. Current defense strategies in online RL rely on active environment interaction, which is infeasible in offline settings. Detection-based methods adapted from other domains also exhibit limited effectiveness in our evaluations. Motivated by these challenges, we propose several preliminary directions for defense:

- Coverage-Aware Data Augmentation: Leveraging generative models to synthesize trajectories in underrepresented regions can reduce distributional mismatch and mitigate poisoning effects.
- Sequence Consistency Checking: Incorporating plausibility-based filtering during training can suppress anomalous transitions and reinforce learning from coherent behavioral patterns.
- Robust Policy Optimization: Enhancing existing algorithms with sequence-level uncertainty regularization may reduce sensitivity to rare or corrupted trajectories.

We hope this work fosters research on secure and coverage-aware offline RL.

7 CONCLUSION

This paper provides an in-depth analysis and systematic empirical study of the robustness and sensitivity of offline RL when faced with data issues with insufficient sequence-level data-policy coverage. Theoretical analysis reveals that insufficient sequence-level coverage exponentially amplifies the upper bound of estimation errors, leading to significant performance degradation in policy learning. We propose a collapsing sequence-level data-policy poisoning attack, thereby unveiling the potential impacts of distributional shift on offline RL performance. The goal is to inspire future researchers to collect high-quality datasets during the online phase and to further explore the design of robust and secure offline RL algorithms.

ACKNOWLEDGMENTS

We would like to thank anonymous reviewers for their constructive comments. This work was supported by the National Natural Science Foundation of China (No.62406086, No.62272127), the Joint Funds of the National Natural Science Foundation of China (No.U22A2036, No.U21B2019), Basic Research Projects of the Central Universities and Colleges (3072025ZN0602), and Natural Science Foundation of Heilongjiang Province of China (No.TD2022F001).

References

Rishabh Agarwal, Dale Schuurmans, and Mohammad Norouzi. An optimistic perspective on offline reinforcement learning. In *International Conference on Machine Learning*, 2019.

Shmuel Bar-David, Itamar Zimerman, Eliya Nachmani, and Lior Wolf. Decision S4: efficient sequence-based RL via state spaces layers. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Yevgen Chebotar, Karol Hausman, Yao Lu, Ted Xiao, Dmitry Kalashnikov, Jacob Varley, Alex Irpan, Benjamin Eysenbach, Ryan C. Julian, Chelsea Finn, and Sergey Levine. Actionable models: Unsupervised offline reinforcement learning of robotic skills. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, pages 1518–1528. PMLR, 2021.

Jinglin Chen and Nan Jiang. Information-theoretic considerations in batch reinforcement learning. In *International Conference on Machine Learning*, 2019.

Christopher Diehl, Timo Sebastian Sievernich, Martin Krüger, Frank Hoffmann, and Torsten Bertram. Uncertainty-aware model-based offline reinforcement learning for automated driving. *IEEE Robotics and Automation Letters*, 8(2):1167–1174, 2023.

Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López, and Vladlen Koltun. Carla: An open urban driving simulator. In *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, pages 1–16. PMLR, 2017.

Justin Fu, Aviral Kumar, Ofir Nachum, G. Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *ArXiv*, abs/2004.07219, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In Marc'Aurelio Ranzato, Alina Beygelzimer, Yann N. Dauphin, Percy Liang, and Jennifer Wortman Vaughan, editors, *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, pages 20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, pages 2052–2062. PMLR, 2018.

Chen Gong, Zhou Yang, Yunpeng Bai, Jieke Shi, Junda He, Kecen Li, Bowen Xu, Sinha Arunesh, Xinwen Hou, David Lo, et al. Baffle: Hiding backdoors in offline reinforcement learning datasets. In *2024 IEEE Symposium on Security and Privacy (SP)*, pages 218–218. IEEE Computer Society, 2024.

Anil K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognit. Lett.*, 31:651–666, 2008.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020a.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33:21810–21823, 2020b.

Jernej Kos and Dawn Song. Delving into adversarial attacks on deep policies. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Workshop Track Proceedings*, 2017.

Aviral Kumar, Justin Fu, G. Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Neural Information Processing Systems*, 2019.

Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020a.

Aviral Kumar, Aurick Zhou, G. Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 1179–1191, 2020b.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020c.

Gukyeong Kwon, Mohit Prabhushankar, Dogancan Temel, and Ghassan AlRegib. Backpropagated gradient representations for anomaly detection. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXI*, volume 12366, pages 206–226. Springer, 2020.

Byung-Jun Lee, Jongmin Lee, Peter Vrancx, Dongho Kim, and Kee-Eung Kim. Batch reinforcement learning with hyperparameter gradients. In *International Conference on Machine Learning*, 2020.

Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17(1):39:1–39:40, 2016.

Sergey Levine, Aviral Kumar, G. Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *ArXiv*, 2020.

Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. Tactics of adversarial attack on deep reinforcement learning agents. In Carles Sierra, editor, *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 3756–3762. ijcai.org, 2017.

Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Zhu. Policy poisoning in batch reinforcement learning and control. *Advances in Neural Information Processing Systems*, pages 14543–14553, 2019.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin A. Riedmiller, Andreas Kirkeby Fidjeland, Georg Ostrovski, Stig Petersen, Charlie Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Rémi Munos. Error bounds for approximate policy iteration. In *International Conference on Machine Learning*, 2003.

Rafael Figueiredo Prudencio, Marcos R.O.A. Maximo, and Esther Luna Colombini. A survey on offline reinforcement learning: Taxonomy, review, and open problems. *IEEE transactions on neural networks and learning systems*, PP, 2022.

Xinghua Qu, Zhu Sun, Yew-Soon Ong, Abhishek Gupta, and Pengfei Wei. Minimalistic attacks: How little it takes to fool deep reinforcement learning policies. *IEEE Trans. Cogn. Dev. Syst.*, 13(4):806–817, 2021.

Amin Rakhsha, Goran Radanovic, Rati Devidze, Xiaojin Zhu, and Adish Singla. Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 7974–7984. PMLR, 2020.

Anshuka Rangi, Haifeng Xu, Long Tran-Thanh, and Massimo Franceschetti. Understanding the limits of poisoning attacks in episodic reinforcement learning. In Luc De Raedt, editor, *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI 2022, Vienna, Austria, 23-29 July 2022*, pages 3394–3400. ijcai.org, 2022.

Tankred Saanum, Noémi Élteto, Peter Dayan, Marcel Binz, and Eric Schulz. Reinforcement learning with simple sequence priors. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023.

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Maxwell Standen, Junae Kim, and Claudia Szabo. Sok: Adversarial machine learning attacks and defences in multi-agent reinforcement learning. *arXiv preprint arXiv:2301.04299*, 2023.

Jianwen Sun, Tianwei Zhang, Xiaofei Xie, Lei Ma, Yan Zheng, Kangjie Chen, and Yang Liu. Stealthy and efficient adversarial attacks against deep reinforcement learning. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 5883–5891. AAAI Press, 2020.

Yanchao Sun, Da Huo, and Furong Huang. Vulnerability-aware poisoning mechanism for online RL with unknown dynamics. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Yanchao Sun, Ruijie Zheng, Yongyuan Liang, and Furong Huang. Who is the strongest enemy? towards optimal and efficient evasion attacks in deep RL. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT press, 2018.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. *2012 IEEE/RSJ*

International Conference on Intelligent Robots and Systems, pages 5026–5033, 2012.

Zhendong Wang, Jonathan J. Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023.

Yifan Wu, G. Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *ArXiv*, abs/1911.11361, 2019.

Chien-Min Yu, Ming-Hsin Chen, and Hsuan-Tien Lin. Learning key steps to attack deep reinforcement learning agents. *Machine Learning*, 112(5):1499–1522, 2023.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y. Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: model-based offline policy optimization. In Hugo Larochelle, Marc'Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan D. Liu, Duane S. Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020a.

Huan Zhang, Hongge Chen, Duane S. Boning, and Cho-Jui Hsieh. Robust reinforcement learning on state observations with learned optimal adversary. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*, volume abs/2101.08452. OpenReview.net, 2021.

Xuezhou Zhang, Yuzhe Ma, Adish Singla, and Xiaojin Zhu. Adaptive reward-poisoning attacks against reinforcement learning. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, pages 11225–11234. PMLR, 2020b.

Collapsing Sequence-Level Data-Policy Coverage via Poisoning Attack in Offline Reinforcement Learning (Supplementary Material)

Xue Zhou[1] Dapeng Man[1] Chen Xu[*1] Fanyi Zeng[1] Tao Liu[1] Huan Wang[1] Shucheng He[1] Chaoyang Gao[1] Wu Yang[1]

[1]College of Computer Science and Technology, Harbin Engineering University, China

A CSDPC ALGORITHM

Algorithm 1 describes the CSDPC attack process. The algorithm takes inputs that include the clean offline RL dataset D, the number of clusters k, the length l of the sequence, the perturbation η, and the poisoning rate ρ. It produces a poisoned dataset D'. Output poisoned dataset D'.

Algorithm 1 CSDPC Attack Workflow

```
Input: $(s, a)$: state-action pairs in clean offline RL dataset $D$, $k$: number of clusters, $l$: length of sequences, $\rho$: the poisoning rate
Output: Poisoned dataset $D'$
Perform k-means clustering on all $(s, a)$ in dataset $D$ to partition the state-action space into $k$ clusters
for each $(s, a)$ in $D$ do
    Assign a cluster label based on the k-means result
end for
Initialize an empty dictionary for sequences cluster combinations and counts
for each sequence of length $l$ in $D$ starting at time step $t$ do
    Record the sequence of cluster labels for the sequence $\tau_t = u_t, u_{t+1}, \ldots, u_{t+l}$
    Remove consecutive duplicate labels and obtain the decision pattern $p$
    Increment the count of the decision pattern in the dictionary
end for
Identify cluster combinations with the lowest counts as critical sequences set $\mathcal{P}$ based on $\rho$
for each sequence in $\mathcal{P}$ do
    for each $(s, a)$ in the sequence do
        Apply perturbations to $(s, a)$ to change the decision pattern to the most frequent decision pattern
    end for
end for
Combine all modified and unmodified sequences to form the poisoned dataset $D'$
return $D'$
```

B DETAILED EXPERIMENTAL SETTINGS

B.1 ENVIRONMENT AND DATASETS

Our experiments were conducted on a computer with dual 12-core Intel(R) Xeon(R) CPUs (32GB RAM) and NVIDIA 3090. For a comprehensive evaluation of our attack framework, we selected four complex continuous tasks from the offline environment D4RL [Fu et al., 2020]: Walker2D, Hopper, and Half-Cheetah in the MuJoCo robot simulator [Todorov et al.,

Table 8: The hyperparameter settings for algorithms.

Parameters	BCQ	BEAR	CQL	BC
Optimizer	Adam	Adam	Adam	Adam
Critic Network Learning Rate	0.001	0.003	0.003	-
Actor Network Learning Rate	0.001	0.001	0.001	-
VAE Learning Rate	0.001	0.003	-	-
Batch Size	100	256	256	100
λ	0.75	-	-	-
Critic Network Hidden Units	[400, 300]	-	[256, 256, 256]	-
Actor Network Hidden Units	[400, 300]	-	[256, 256, 256]	-
VAE Encoder Hidden Units	[750, 750]	[750, 750]	-	-
VAE Decoder Hidden Units	[750, 750]	[750, 750]	-	-
γ	0.99	0.99	0.99	1
Activation Function	ReLU	ReLU	ReLU	ReLU
α Learning Rate	-	0.001	-	0.001
α Threshold	-	0.05	-	-

2012], and Carla-Lane autonomous driving tasks in the Carla simulator [Dosovitskiy et al., 2017], which are closer to real-world scenarios.

The MuJoCo tasks require the agent to move forward rapidly in different forms. Each dataset is medium-level and comprises 1 million steps. The Carla-Lane task aims to drive a car smoothly and swiftly forward, with a dataset size of 100,000 pieces.

B.2 EXPERIMENTAL PARAMETERS

In the Mujoco environment, the random seed is set to 0, in the Carla environment, it is set to 5. The hyperparameter settings for the algorithms used in the experiments are presented in Table 8.

B.3 OFFLINE RL ALGORITHMS

To evaluate how offline RL algorithms perform under our poisoning framework, we employed three algorithms for training agent: Batch-Constrained Q-learning (BCQ) [Fujimoto et al., 2018], Batch-Ensemble Actor-Critic with Retrace (BEAR) [Kumar et al., 2019], Conservative Q-Learning (CQL) [Kumar et al., 2020b] and Behavioural Cloning (BC) [Sutton and Barto, 2018]. We use the official open-source code of these algorithms and follow the settings by D4RL [Fu et al., 2020].

C MORE RESULTS

C.1 THE EFFECTIVENESS OF CSDPC ATTACK

When the poisoning rate is 5%, the effectiveness of CSDPC attack is as shown in Table 9.

C.2 ADDITIONAL COMPARATIVE RESULTS

Although the online attack method cannot be applied to offline RL due to the need to interact with the environment in real-time to obtain training parameters, however, in order to more fully evaluate the performance of the CSDPC method, we

Table 9: The values are the ACR obtained by the agent in various environments, the values in parentheses are the AER in poisoned agents, and $\eta = 0.05$, $\rho = 5\%$.

Algorithms	Walker2D			Hopper			Half			Carla		
	Clean	Raw	Feature	Clean	Raw	Feature	Clean	Raw	Feature	Clean	Raw	Feature
CQL	3132	322(90%)	180(94%)	3158	180(94%)	70(98%)	4822	573(88%)	437(91%)	191	39(79%)	24(87%)
BEAR	2593	55(98%)	40(98%)	2119	111(95%)	66(97%)	4290	382(91%)	337(92%)	89	14(84%)	8(91%)
BCQ	2341	113(95%)	69(97%)	2823	238(92%)	113(96%)	4694	676(86%)	576(88%)	466	79(83%)	53(89%)
BC	744	81(89%)	39(97%)	3450	229(93%)	99(97%)	4017	401(90%)	338(92%)	384	69(82%)	56(85%)
Average	2203	143(93%)	82(96%)	2613	190(93%)	87(97%)	4456	508(89%)	421(91%)	283	50(82%)	35(88%)
	Raw Data AER 89%						Feature Data AER 93%					

Table 10: AER of the poisoned agents under different attacks. The best results are indicated in bold.

Environments	Attack Methods	RL	Target Algorithms	η	AER
Hopper	RS [Zhang et al., 2020a]	online	PPO	0.75	75%
	SA-RL[Zhang et al., 2020a]	online	PPO	0.75	80%
	PA-AD [Sun et al., 2022]	online	PPO	0.75	**95%**
	BAFFLE [Gong et al., 2024]	offline	CQL	-	48%
	CSDPC (Ours)	offline	CQL	0.05	**97%**
Half-Cheetah	RS [Zhang et al., 2020a]	online	PPO	0.15	93%
	SA-RL [Zhang et al., 2020a]	online	PPO	0.15	100%+
	PA-AD [Sun et al., 2022]	online	PPO	0.15	**100%+**
	BAFFLE [Gong et al., 2024]	offline	CQL	-	59%
	CSDPC (Ours)	offline	CQL	0.05	91%
Walker2D	RS [Zhang et al., 2020a]	online	PPO	0.05	70%
	SA-RL [Zhang et al., 2020a]	online	PPO	0.05	76%
	PA-AD [Sun et al., 2022]	online	PPO	0.05	82%
	BAFFLE [Gong et al., 2024]	offline	CQL	-	55%
	CSDPC (Ours)	offline	CQL	0.05	**96**%

provide a succinct comparison between CSDPC attack and previously established online and offline RL attack methods. The comparative results are summarized in Table 10. The results demonstrate that the effectiveness of our attack method is optimized when the size of perturbations is identical or nearly so.

In the Half-Cheetah environment, the magnitude of perturbations employed by other online RL attack methods is threefold that of our CSDPC attack. This substantial difference in perturbation magnitude could account for the capacity of these methods to not only markedly degrade the performance of the agents but also, induce penalties.