Sample-Efficient Reinforcement Learning via Counterfactual-Based Data Augmentation

Chaochao Lu[*1,3], Biwei Huang[*2], Ke Wang[6], Jośe Miguel Hernández-Lobato[1,4,5], Kun Zhang[2], and Bernhard Schölkopf[3]

Abstract

Reinforcement learning (RL) algorithms usually require a substantial amount of interaction data and perform well only for specific tasks in a fixed environment. In some scenarios such as healthcare, however, usually only few records are available for each patient, and patients may show different responses to the same treatment, impeding the application of current RL algorithms to learn optimal policies. To address the issues of mechanism heterogeneity and related data scarcity, we propose a data-efficient RL algorithm that exploits structural causal models (SCMs) to model the state dynamics, which are estimated by leveraging both commonalities and differences across subjects. The learned SCM enables us to counterfactually reason what would have happened had another treatment been taken. It helps avoid real (possibly risky) exploration and mitigates the issue that limited experiences lead to biased policies. We propose counterfactual RL algorithms to learn both population-level and individual-level policies. We show that counterfactual outcomes are identifiable under mild conditions and that Q-learning on the counterfactual-based augmented data set converges to the optimal value function. Experimental results on synthetic and real-world data demonstrate the efficacy of the proposed approach.

1 Introduction

Over the last few years, reinforcement learning (RL) has been successfully applied to challenging problems such as playing Go [1] and Atari games [2]. Key factors of the success include a substantial amount of interaction data and that the ongoing tasks are well-designed in a fixed environment. However, these factors do not always hold in real-world scenarios. A case in point is the RL formulation of healthcare, which aims to optimize sequential treatments to achieve recovery [3, 4]. Healthcare data usually have the properties that (1) only a few records are available for each patient and no further exploration can be performed, and that (2) patients may show different responses to identical treatments. These two properties impede the application of most RL algorithms to learn optimal policies.

Regarding handling the issue of data scarcity, model-based approaches tend to be more sample efficient and allow better interpretability than model-free ones. However, model-based approaches have problems when dealing with complex dynamics. Several hybrid approaches have been considered to mitigate these limitations. For example, model-based value expansion (MVE [5]) incorporates a fixed depth of predictive models of system dynamics into model-free value function estimation, and later stochastic ensemble value expansion (STEVE [6]) further mitigates the bias due to model mis-specification by dynamically interpolating various horizon lengths. Besides, a recent approach based on counterfactually-guided policy search [7] models the dynamics with a pre-defined structural causal model (SCM) and performs probabilistic counterfactual reasoning to generate alternative

[1]University of Cambridge, [2]Carnegie Mellon University, [3]Max Planck Institute for Intelligent Systems, [4]Microsoft Research Cambridge, [5]Alan Turing Institute, [6]Imperial College London, [*]Equal Contribution. Correspondence Authors: cl641@cam.ac.uk, biweih@andrew.cmu.edu.

Offline Reinforcement Learning Workshop at Neural Information Processing Systems, 2020.

outcomes under counterfactual actions. Its implementation assumes that the ground-truth transition and reward kernels are all given, which may not be realistic in some cases.

Regarding heterogeneity across subjects or environments, a common strategy is to use meta-RL. For instance, context-based meta-RL methods adapt to new tasks by aggregating experience into a latent representation on which the policy is conditioned [8, 9, 10, 11]. In practice, meta-RL is usually hard to train, and during training, it may require large amounts of data drawn from a large set of distinct tasks, exacerbating the problem of sample efficiency.

To address the issues of mechanism heterogeneity and related data scarcity, we propose a sample-efficient RL algorithm, leveraging the following properties:

1. Although the treatment effect may be different across individuals, a large proportion may still show a similar trend. We leverage commonalities and take into account variations across individuals to achieve more reliable estimation.
2. We leverage structural causal models (SCMs) to model the dynamic process. For generality, we do not put hard constraints on the functional class of causal mechanisms or data distributions. Moreover, to take into account mechanism heterogeneity, we include a variable θ_C in the causal system explicitly, to characterize hidden factors which change across individuals.
3. Given the SCM, accordingly we perform counterfactual reasoning by following Pearl's procedure [12]. Counterfactual reasoning is the process of evaluating conditional claims about alternate possibilities and their consequences. For example, in our case, we can leverage it to infer that given the observed state-action tuple $\langle S_t = s_t, A_t = a, S_{t+1} = s_{t+1}\rangle$, what would have happened had we performed a'. This helps avoid real (possibly risky) exploration, which is often infeasible in real-world scenarios, and mitigates the problem that limited experiences lead to biased policies.

Contributions: We propose a practically useful and theoretically sound approach to tackling mechanism heterogeneity and data scarcity in RL, by counterfactual data augmentation. 1) On the technical side, we achieve flexible counterfactual reasoning in the general case, by adopting neural networks for function approximation of the SCM with no hard restrictions on data distribution or causal mechanisms, and use it for data augmentation to address the issues in RL. 2) We explicitly model the changing factors across subjects to achieve personalized policies, as well as a general policy over the population. 3) Theoretically, we show that the counterfactual outcome is identifiable under rather weak conditions. Furthermore, we show that Q-learning on the counterfactual-based augmented data set converges to the optimal value function.

2 Preliminaries

In this section, we briefly introduce structural causal models and counterfactual reasoning which will be used throughout the paper.

2.1 Structural Causal Models

Let $\mathbf{Y} = \{Y_1, \cdots, Y_n\}$ be a set of n observed variables. A structural causal model consists of a set of equations of the form [12]:

$$Y_i = f_i(X_i, U_i), \tag{1}$$

for $i = 1, \cdots, n$, where X_i stands for the set of parents of Y_i, i.e., a subset of the remaining variables in $\mathbf{Y}$ that directly cause Y_i, and U_i represents disturbances (noises) due to omitted factors. Each of the functions f_i represents a causal mechanism that determines the value of Y_i from the causes and the noise term on the right side. The functional characterization in Eq. (1) provides a convenient language for specifying how the resulting distribution would change in response to interventions.

2.2 Counterfactual Reasoning

Suppose we performed action a in state s_t and observed s_{t+1}. One may be interested in knowing what would have happened had we performed a'. This is a *counterfactual* question. It has been shown that given an SCM, we can then perform counterfactual reasoning [12].

Suppose the SCM in (1) is given, denoted by M, and that we have evidence $Y = y$ and $X = x$ (the subscript i has been omitted for representation convenience). The following steps show how to counterfactually infer Y had we set $X = x'$ [12]:

- Step 1 (abduction): Use evidence $(Y = y, X = x)$ to determine the value of U.
- Step 2 (action): Modify the model, M, by removing the structural equations for the variables in X and replacing them with the function $X = x'$, to obtain the modified model, $M_{x'}$.

- Step 3 (prediction): Use the modified model, $M_{x'}$, and the value of U to compute the counterfactual value of Y.

The counterfactual outcome is usually represented as: $Y_{X=x'}|Y=y, X=x$. Note that in Step 1, we perform deterministic counterfactual, that is, counterfactuals pertaining to a single unit of the population, where the value of U is determined.

3 Counterfactual RL Using SCMs

We now propose data-efficient RL algorithms, by leveraging SCMs and its corresponding counterfactual-based data augmentation to handle the issues of data scarcity and mechanism heterogeneity. In this section, we first propose CounTerfactual Reinforcement Learning of a general policy, denoted by CTRL$_g$, aiming for a policy over the population. Next, we propose CounTerfactual Reinforcement Learning of personalized policies, denoted by CTRL$_p$, providing personalized policies for each individual or each automatically determined group.

3.1 CTRL$_g$: Estimation of a General Policy

When examining whether a treatment is effective and should be adopted as a standard, it is first essential to understand its effect over the population. In the following, we focus on the estimation of a general policy for the population.

We assume that the state S_{t+1} satisfy the SCM

$$S_{t+1} = f(S_t, A_t, U_{t+1}), \tag{2}$$

where f represents the causal mechanism, A_t the action at time t, and U_{t+1} the noise term, which is independent of $(S_t; A_t)$. To estimate a general policy, we do not consider the variability across individuals.

Given observed triplets $\langle S_t, A_t, S_{t+1}\rangle$ from individuals, for $t = 1, \cdots, T$, the first problem is how to efficiently estimate the causal mechanism f. To achieve generality, we do not specify a particular functional class of the causal mechanism, e.g., linear relations [13], nonlinear relations with additive noise [14], or the causal model with further post-nonlinear transformations [15]. Instead, we use a generative adversarial framework to learn f, by minimizing the discrepancy between real data and generated data. In addition, to achieve counterfactual-based data augmentation, we need to estimate the value of the noise term at each time point in (2).

To achieve the two goals, estimating f and noise values simultaneously, we cast the learning of both an inference machine (encoder) and a deep generative model (decoder) in a generative adversarial network (GAN)-like adversarial framework, called Bidirectional Conditional GAN (BiCoGAN [16]). Specifically, the BiCoGAN contains two parts: one is a generative model mapping from $\langle S_t, A_t, U_{t+1}\rangle$ to S_{t+1}, and the other is an inference mapping from S_{t+1} to $\langle S_t, A_t, U_{t+1}\rangle$. The discriminator is trained to discriminate between joint samples from encoder distribution

$$P(S_{t+1}, \hat{S}_t, \hat{A}_t, \hat{U}_{t+1}) = P(S_{t+1})P(\hat{S}_t, \hat{A}_t, \hat{U}_{t+1}|S_{t+1})$$

and decoder distribution

$$P(\hat{S}_{t+1},S_t,A_t,U_{t+1}) = P(S_t,A_t,U_{t+1})P(\hat{S}_{t+1}|S_t,A_t,U_{t+1}),$$

where $\hat{S}$, $\hat{A}_t$, and $\hat{U}_{t+1}$ denote the estimations of S, A_t, and U_{t+1}, respectively. The decoder and the encoder learn conditionals $P(\hat{S}_{t+1}|S_t, A_t, U_{t+1})$ and $P(\hat{S}_t, \hat{A}_t, \hat{U}_{t+1}|S_{t+1})$, respectively, to fool the discriminator, with objective function:

$$\min_{G,E}\max_{D} V(D,G,E) = \min_{G,E}\max_{D}\Big\{\underbrace{\mathbb{E}_{S_{t+1}\sim P_{\text{data}}(S_{t+1})}[\log D(E(S_{t+1}), S_{t+1})]}_{\text{Encoder}}$$
$$+\underbrace{\mathbb{E}_{\ddot{Z}_t\sim P_{\ddot{Z}_t}(\ddot{Z}_t)}[\log(1-D(G(\ddot{Z}_t),\ddot{Z}_t))]}_{\text{Decoder}} + \underbrace{\lambda\mathbb{E}_{(S_t,A_t,S_{t+1})\sim P_{\text{data}}(S_t,A_t,S_{t+1})}[R((S_t,A_t),E(S_{t+1})]}_{\text{Regularizer}}\Big\},$$

where $\ddot{Z}_t = (S_t, A_t, U_{t+1})$, D denotes a discriminator network, G a generator network, E an encoder network, and R is a regularizer (with hyperparameter λ) to prevent overfitting the estimation error of (S_t, A_t), which helps to better encode extrinsic factors. A graph illustration is given in Figure 1a.

After learning the SCM, including the causal mechanism $\hat{f}$ and noise values $\hat{u}_{t+1}$, for $t = 1, \cdots, T$, we can counterfactually reason what would have happened if another treatment had been taken. Suppose at time $t+1$, we have $\langle S_t = s_t, A_t = a, S_{t+1} = s_{t+1}\rangle$. We want to know what would have been the next state, if we had taken the action a'. Practically, this can be achieved by feeding s_t, a', and $\hat{u}_{t+1}$ into the learned generator network G, and the output is the counterfactual outcome s'_{t+1}.

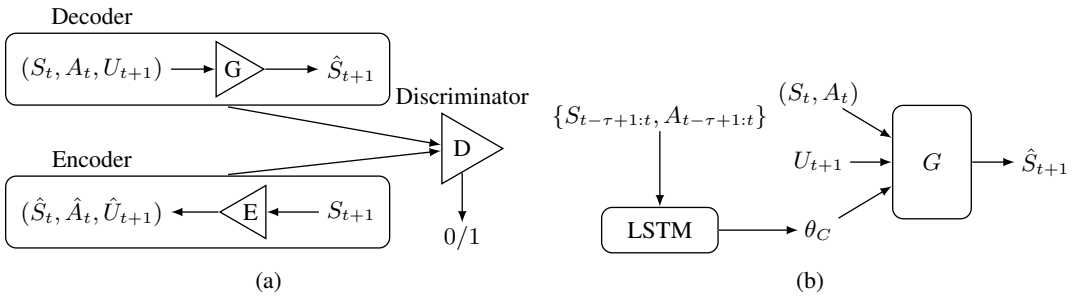


Figure 1: (a) Generator G, Encoder E, and Discriminator D in CTRL$_g$. (b) Generator in CTRL$_p$.

The alternative actions a' are chosen in the following way. Suppose the action space is $\mathcal{A} \sim P_{\mathcal{A}}$ with support $[b^-, b^+]$, which can be either discrete or continuous; for example, in healthcare, whether surgery is performed is a binary variable, while a drug dosage may be continuous. We uniformly sample from $[b^-, b^+]$ to generate alternative actions a' and accordingly estimate the counterfactual outcome s'_{t+1}. We denote by $\tilde{\mathbb{D}}$ the augmented data set after including the data from counterfactual reasoning. Counterfactually reasoning about the effects of alternative actions helps avoid possibly risky exploration and mitigates the problem that limited experience leads to biased policies in RL.

Remark 1 (SCMs vs. standard model-based approaches). SCMs directly involve noise term U, which makes counterfactual reasoning possible. In principle, standard model-based approaches (e.g., [17, 18]) do not have U as input and cannot directly produce counterfactual outcomes–counterfactual reasoning is not possible without a causal model. Moreover, our approach nicely benefits from the power of the recently developed GAN-type methods in capturing the property of high-dimensional (conditional) distributions and generating new random samples. By contrast, the dynamics in standard model-based approaches are usually parameterized with a Gaussian distribution or a Gaussian mixture distribution. Although in the limit of infinitely many components, Gaussian mixture can fit any distribution, this becomes unpractical if the numbers get too large.

Remark 2 (causal reasoning vs. standard Monte-Carlo simulations). Standard Monte-Carlo simulations are not able to directly perform (individual-level) counterfactual reasoning, and hence cannot achieve our goal. As pointed out by Pearl [12], in counterfactual reasoning one has to exploit a causal model, and then he/she is able to infer the specific property of the considered individual, related to the noise term U, and then exploit it to derive what would have happened if we had performed alternative action a' (for the same individual). Monte-Carlo simulations can only produce random samples at the population (not individual) level.

3.2 CTRL$_p$: Estimation of Personalized Policies

In healthcare, patients can exhibit different responses to identical treatments. Therefore, one should care not only about the treatment effect for the general population, but also the response of each individual or each properly divided group. In the following, we propose counterfactual RL of personalized policies (CTRL$_p$), by taking into account variabilities across individuals/groups and leveraging commonalities to achieve statistically reliable estimation.

We use a variable θ_C to explicitly take into account hidden factors, whose value may vary across individuals, where C denotes the subject index. Thus, for an individual, we assume that the state S_{t+1} satisfies the SCM

$$S_{t+1} = f(S_t, A_t, \theta_C, U_{t+1}), \tag{3}$$

where f represents the overall family of mechanisms, and θ_C captures factors that depends on the subject or group.

To capture the variation (i.e., estimate the value of θ_C), we segment the data sequence of each subject using sliding windows of size τ, resulting in triplets $\{\langle S_{t-i+1}, A_{t-i+1}, S_{t-i+2}\rangle\}_{i=1}^{\tau}$. At each time t, we exploit individual-specific information from the sequence $\{S_{t-\tau+1:t}, A_{t-\tau+1:t}\}$, by leveraging Long-Short Term Memory (LSTM) [19]. The LSTM output $\hat{\theta}_C$ acts as a new input to the generator G. Note that we constrain the same individual to have the same value for θ_C. Figure 1b illustrates the generator G.

Similar to CTRL$_g$, CTRL$_p$ is also estimated with BiCoGAN, with the difference that there is a latent variable θ_C, learned with an LSTM network, as a new conditioning variable, i.e., conditioning

variables $\ddot{Z} = (S_t, A_t, \theta_C, U_{t+1})$. All parameters in CTRL$_p$ are learned simultaneously in an adversarial manner.

After learning the SCM, including f, θ_C, and U_{t+1}, we divide individuals into groups by applying k-means clustering to the estimated values of $\hat{\theta}_C$. We use the estimated centroids from k-means as a new $\tilde{\theta}_C$, and hence $\tilde{\theta}_C$ is constant within each group but varies across groups. We can then perform counterfactual reasoning on each group of individuals, as described in Section 3.1, resulting in an augmented dataset $\check{\mathbb{D}}_i$ for the i-th group.

3.3 Identifiability

Given triplets $\langle S_t = s_t, A_t = a, S_{t+1} = s_{t+1} \rangle$, it is important to show whether the counterfactual outcome is identifiable. Without this guarantee, the output of the method may be different from the true counterfactual outcome. Surprisingly, the following theorem shows that without any hard constraints on the functional form f and the noise distribution in the SCM, the derived counterfactual outcome is correct under weak assumptions. This makes counterfactual reasoning generally possible.

Theorem 1. *Suppose S_{t+1} satisfies the following structural causal model:*

$$S_{t+1} = f(S_t, A_t, U_{t+1}),$$

where $U_{t+1} \perp\!\!\!\perp (S_t; A_t)$, and we assume that f (which is unknown) is smooth and strictly monotonic in U_{t+1} for fixed values of S_t and A_t. Suppose we have observed $\langle S_t = s_t, A_t = a, S_{t+1} = s_{t+1} \rangle$. Then for the counterfactual action $A_t = a'$, the counterfactual outcome

$$S_{t+1, A_t = a'} | S_t = s_t, A_t = a, S_{t+1} = s_{t+1} \tag{4}$$

is identifiable.

Note that the above theorem naturally holds for the specific SCM in Eq. (3), since for an individual $C = c$, θ_c is fixed and thus, equivalently, Eq. (3) can be written as $S_{t+1} = f_c(S_t, A_t, U_{t+1})$. The monotonicity condition on f with respect to U_{t+1} guarantees that the noise term is recoverable. Consider extreme cases, nonlinear causal models with additive noise or multiplicative noise, where the effect variable is always strictly monotonically increasing in the noise, rendering the noise recoverable from the cause and effect. Moreover, the above theorem holds no matter whether the state space and the action space are continuous or discrete.

Furthermore, note that although the function f and the probabilistic distribution $P(U_{t+1})$ are not uniquely identifiable from the collected data without strong constraints on the functional class of f and on the data distributions [20], surprisingly, the constructed counterfactual outcome does not depend on which f and $P(U_{t+1})$ we choose, provided that f is strictly monotonic in U_{t+1}. In experiments, this strict monotonicity can be easily implemented through monotonic multi-layer perceptron network [21], in which positive signs of the weights are guaranteed by introducing their exponential form [22].

Remark 3. Previously, the identifiability of counterfactual quantities was only shown in the case where both cause X and effect Y are binary and Y is monotonic relative to X [12]. Later, [23] discusses the case with categorical variables. Note that in previous work, the monotonicity is with respect to X (equivalent to A_t in Eq.(2)), while in our theorem, it is relative to the noise term U. Interestingly, we find that by requiring monotonicity in U, we can show identifiability for more general data types and causal mechanisms. Moreover, note that here we only care about the identifiability of the counterfactual outcomes, which does not require identifiability of the SCM.

4 Deep Q-Networks

After estimating the dynamics model and generating an augmented data set $\tilde{\mathbb{D}}$ by counterfactual reasoning, we are now ready to learn policies on $\tilde{\mathbb{D}}$ to maximize future rewards. To this end, we use the Dueling Double-Deep Q-Network (D3QN) [24], a variant of Deep Q-Networks (DQNs) [25].

A simple DQN may suffer from shortcomings that Q-values are often overestimated, which leads to inaccurate predictions and sub-optimal policies [24]. This problem can be mitigated by using Double-Deep Q-Networks [26], where the target Q values are determined using actions found through a feed-forward pass on the main network, instead of being estimated directly from the target network.

In addition, a high Q value may be due to (1) a patient's positive state, e.g., being near discharge, or (2) a treatment that is taken at that time step [24]. For general applicability of the learned policy, it is important to distinguish between these two cases. To achieve this, one may use Dueling Q-Networks [27], where $\mathcal{Q}(s, a)$ is split into separate value and advantage streams, representing the quality

of the current state and the quality of the chosen action, respectively. Leveraging the advantages from Double-Deep Q-Networks and Dueling Q-Networks, the final network is a fully connected Dueling Double-Deep Q-Network. Algorithm 1 gives the entire procedure of policy learning via counterfactual-based data augmentation.

The following theorem shows that on the counterfactually augmented data set, Q-learning converges to the optimal value function.

Theorem 2. *Given the transition dynamics, Q-learning on the counterfactually augmented data set converges with probability one to the optimal value function Q^*, as long as the state and action spaces are finite, and the learning rate α_t satisfies $\sum_t \alpha_t = \infty$ and $\sum_t \alpha_t^2 < \infty$.*

Remark 4. In this paper, we used deterministic counterfactuals (see Section 2.2), where the value of U_{t+1} can be determined, with which we show that the estimation is consistent. Instead, CF-PE [7] performs probabilistic counterfactuals, where the value of U_{t+1} is sampled from the posterior distribution $P(U_{t+1}|\mathbb{D})$. CF-PE is unbiased, if there is no model mismatch, but it is not guaranteed to converge to the optimal value function. Moreover, if U_{t+1} is directly sampled from the prior distribution $P(U_{t+1})$, which is usually the case in a probabilistic transition model, the estimation is even biased [7].

5 Experimental Results

To evaluate the proposed approaches CTRL$_g$ and CTRL$_p$, we applied them to a modified classical control problem and a real-world healthcare dataset. More experiments and details about model architectures and implementation are given in Appendix.

5.1 Results on Classical Control Problems

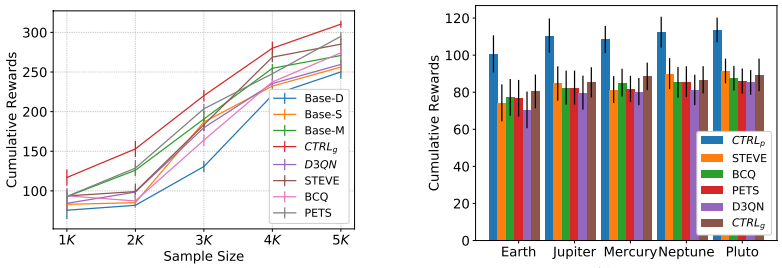
We first evaluated the performance of our proposed methods on the cartpole environment in OpenAI gym, a benchmark for classical control tasks. It has continuous states with dimension $d_s = 4$ and discrete actions with dimension $d_a = 1$. To increase the complexity and make the environment stochastic, we extended the original two discrete actions (i.e., $a = 0, 1$) to eleven actions (i.e., $a = 0, 0.1, \ldots, 0.9, 1$) and added 5% Gaussian noise to both states and actions.

To train our models and evaluate their performance, we created two datasets: **SD**: Simple dataset with fixed gravity ($g_{\text{earth}} = 9.8$) and **HD**: Hybrid dataset with five different gravities ($g_{\text{jupiter}} = 24.79$, $g_{\text{earth}} = 9.8$, $g_{\text{mercury}} = 3.7$, $g_{\text{neptune}} = 11.15$, and $g_{\text{pluto}} = 0.62$). In **SD**, we collected 250 trials by applying random actions and put the collected data in five data subsets with the number of trials $n_{\text{trial}} = 50, 100, ..., 250$. Each trial has 20 consecutive steps and each step contains $\langle s_t, a_t, s_{t+1}, r_t \rangle$. In **HD**, for each gravity, we generated 50 trials in the same way as that in **SD**. Additionally, we generated 5-time-step sequential data by applying sliding windows along each trajectory for CTRL$_p$.

Comparisons to Baselines and SOTA We first compared the proposed CTRL$_g$ with three well-known baselines and three state-of-the-art (SOTA) approaches in terms of sample efficiency on **SD**. The three baselines are summarised as follows (more details can be found in Appendix). (1) Base-D (deterministic dynamics model): The next state S_{t+1} is determined by current state S_t and current action A_t with $S_{t+1} = f(S_t, A_t)$. (2) Base-S (probabilistic dynamics model with unimodal

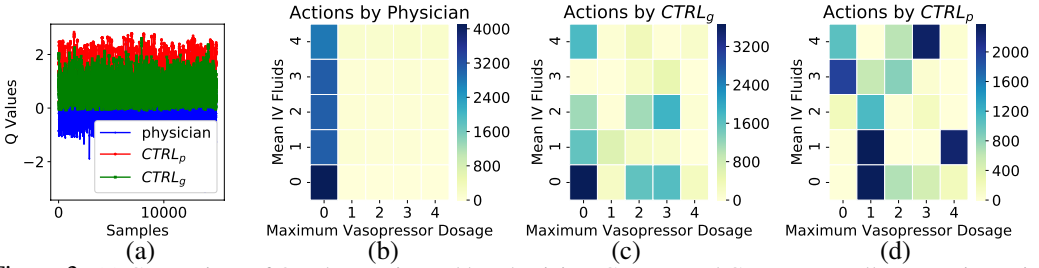
Algorithm 1 Policy Learning via Counterfactual-Based Data Augmentation

1. **Input:** observed triplets $\langle S_t, A_t, S_{t+1} \rangle$ from individuals, for $t = 1, \cdots, T$.
2. **Estimation of a general policy:**
   - 2.1. Estimate the SCM given in (2) with BiCoGAN.
   - 2.2. Generate counterfactual data given alternative actions, according to the estimated SCM. Denote the counterfactually augmented data set by $\tilde{\mathbb{D}}$.
   - 2.3. Perform D3QN learning on $\tilde{\mathbb{D}}$.
3. **Estimation of personalized policies:**
   - 3.1. Estimate the SCM given in (3) with BiCoGAN, meanwhile with LSTM to characterize individual specific factor θ_C.
   - 3.2. For each individual or each automatically determined group, generate counterfactual data given alternative actions, according to its estimated SCM. Denote the counterfactually augmented data set for the i-th individual by $\check{\mathbb{D}}_i$.
   - 3.2. Perform D3QN learning on $\check{\mathbb{D}}_i$.
4. **Output:** General policy over the population and personalized policies for each individual or each group.



(a) (b)

Figure 2: (a) CTRL$_g$ outperforms three baselines, three state-of-the-art methods, and D3QN on the original data on **SD**. The improvement is obvious, especially when the sample size is small. (b) The general policy CTRL$_g$ is competitive, and the personalized policy CTRL$_p$ outperforms state-of-the-art methods on **HD**.



(a) (b) (c) (d)

Figure 3: (a) Comparison of Q-values estimated by Physician, CTRL$_g$, and CTRL$_p$ over all state-action pairs on the test set. (b-d) Distribution of actions obtained by applying Physician, CTRL$_g$, and CTRL$_p$. The axes labels index the discretized action space, where 0 represents no drug given, and 4 is the maximum dosage.

distribution): The probabilistic dynamics model $P(S_{t+1}|S_t, A_t; \theta)$ is Gaussian, where θ denotes a set of parameters in the dynamics. (3) Base-M (probabilistic dynamics model with multimodal distribution): $P(S_{t+1}|S_t, A_t; \theta)$ is a mixture of Gaussians, implemented by a mixture density network [28]. The three SOTA approaches are STEVE [6], BCQ [29], and PETS [18].

For a fair comparison, all methods are trained on the five subsets in **SD** (with $n_{\text{trial}} = 50, 100, ..., 250$) without exploration in the environment, and all results are reported after training converges. After learning the transition dynamics, we counterfactually generated new data for each and trained a D3QN on a mixture of real data and generated data. For the baseline models, counterfactual reasoning was replaced by generating s_{t+1} via fixing s_t and applying a random a_t. Then we tested the learned policy in the CartPole simulation by performing 10 trials with different random seeds. We also compare with D3QN on the original data, without data augmentation, denoted by D3QN. Figure 2a shows that CTRL$_g$ outperforms all other methods, with the highest cumulative reward in all settings, especially when the sample size is small.

Next, we evaluated the performance of CTRL$_p$ on **HD**. Based on the grouping information obtained by CTRL$_p$, we counterfactually generated more data for each group and then learned an optimal policy on each. At test time, we first estimated the group membership of a given patient, and then applied the optimal group-specific policy. Figure 2b demonstrates that the individual policy learned by CTRL$_p$ is superior to the general policy learned by the SOTA methods in all five environments, as shown by the consistently highest cumulative rewards. Note that the SOTA methods learn a general policy, and only our method is individualized. We did not compare with meta-RL, because typically meta-RL-based approaches are not in the batch off-policy setting.

5.2 Results on MIMIC-III

In this section, we investigated the performance of our approach on the real-world *Medical Information Mart for Intensive Care-III* (MIMIC-III) database with Sepsis-3 [30]. Each patient state consists of 46 variables including laboratory tests, vital signs, and patient demographics. We follow the setting of data preparation and the reward function in [24]. We selected 15,656 patients who have at least five consecutive time steps. For each model, we generated the train, validation, and test sets. More precisely, CTRL$_g$ had three sets of size, 120K, 15K, and 15K, respectively. Since CTRL$_p$ needs the sequences as input, the sizes of the three sets were smaller, corresponding to 80K, 10K, and 10K.

General Treatment with CTRL$_g$ We first learned SCM on the (real) training set. Once the model was learned, we randomly selected a pair of (s_t, s_{t+1}) from the set, and used the learned encoder to estimate the noise value $\hat{u}_t$. We then used s_t and $\hat{u}_t$ to generate $\hat{s}_{t+1}$ by taking a uniformly random action. After training the D3QN on the augmented data set, we estimated the Q-values of the

physician policy as well as that of the learned optimal policy (Q-values have been typically used to evaluate patient care, e.g., [24, 3, 4]). In Figure 3a we can see that the optimal policy trained on a combination of real data and the generated data by CTRL$_g$ achieves a larger estimated Q-value than the physician policy. In other words, the optimal policy trained on our counterfactually-generated data is able to increase survival time. Figure 3(b-c) further shows that the policy learned by CTRL$_g$ differs from the physician policy. While both polices exhibit a similar trend to high usage of IV fluids and lower usage of vasopressors, CTRL$_g$ recommends using more vasopressors and less IV fluids.

Personalized Treatments with CTRL$_p$ We trained CTRL$_p$ on sequential data with five time steps. During training, our model uses k-means on the estimated θ_C to group the patients.[1] Exploiting this grouping information, we pool the data from the patients within the same group, then counterfactually generate new data based on the real data in each group, and finally train a D3QN on a mix of real data and generated data for each group. The resulting optimal policy is specific for each group. During the test phase, given a new patient with a sequence of data, we first estimated which group he or she belongs to, and then applied the corresponding policy of that group. As shown in Figure 3a, applying a personalized policy to each patient leads to a larger average estimated Q-value than directly applying the general policy learned from the whole population data. Figures 3(b-d) present the difference of action statistics over all clusters by applying different policies. It seems that compared to the general policy, the optimal personalized policy on average prefers heavier usage of both vassopressors and IV fluids. Interestingly, Figure 3(c-d) suggest that actions learned by CTRL$_p$ are more diversely distributed than that by CTRL$_g$. This indeed makes sense in that the personalized policy encourages personalized treatments for each patient, which can lead to more diversity. It is also reasonable that the policy learned by CTRL$_g$ is closer to the physician policy than that by CTRL$_p$, because the physician policy also comes from their experience of treatment to the population as CTRL$_g$ does. Although without a real-world experiment we cannot verify the true effect of the learned optimal policy, it may provide guidance for physicians to inform their decisions and do further evaluations.

6 Related Work

Many studies have been trying to address the perceived sample inefficiency of RL. Compared to model-free RL, model-based RL tends to be more sample efficient and allow better interpretability. However, current model-based RL algorithms may converge to suboptimal solutions [32]. The reason is that the learned models may fail to reflect the true process from insufficient interaction data and thus lead to biased policies. One way to mitigate this problem is to incorporate uncertainty into the dynamics model. For example, the deterministic dynamics model has been extended by parameterizing it with a Gaussian distribution or a Gaussian mixture distribution for better generalization [28]. PILCO [17] leverages Gaussian processes to express model uncertainty, and further the model uncertainty is incorporated into planning and policy evaluation. Probabilistic Ensembles with Trajectory Sampling algorithm (PETS) [18] combine uncertainty-aware deep network dynamics models with sampling-based uncertainty propagation.

Recently, an algorithm based on probabilistic counterfactually-guided policy search has been proposed in the POMDP setting [7]. In its implementation, it assumes that the ground-truth transition, observation and reward kernels are all given. Furthermore, it adds uncertainty only on the initial state S_1. Although noise added to the initial state can propagate to the whole system, it is not the true causal process. In contrast, we model and implement the underlying causal process with an SCM in the MDP setting, so the noise appears in the SCM at each time step (see Eq. (2)), representing unmeasured factors influencing S_{t+1}. In addition, the work by [33] leverages counterfactual data in multi-armed bandit problems under heterogeneous conditions, with counterfactual quantities being estimated by active agents empirically. Backtracking models [34] consider predicting the samples that may lead to high-reward states.

7 Conclusion

To address the issue of dynamics heterogeneity and data scarcity common in healthcare, we propose a data-efficient RL algorithm that exploits SCMs to model the state dynamics. The learned SCM enables us to counterfactually reason what would happen had another treatment been taken. It helps avoid real (possibly risky) exploration and mitigates the problem that limited experience leads to biased policies. We provide both a general policy over the population and personalized policies for individuals in automatically identified groups. The proposed methods show promising results on both synthetic and real-world datasets.

[1] More sophisticated clustering methods may lead to improved results [31].

References

[1] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, ⋯, and S. Dieleman. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484, 2016.

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

[3] B. Petersen, J. Yang, W. Grathwohl, C. Cockrell, C. Santiago, G. An, and D. Faissol. Precision medicine as a control problem: Using simulation and deep reinforcement learning to discover adaptive, personalized multi-cytokine therapy for sepsis. *arXiv preprint*, page arXiv:1802.10440, 2018.

[4] O. Gottesman, F. Johansson, J. Meier, J. Dent, D. Lee, S. Srinivasan, et al., and J. Yao. Evaluating reinforcement learning algorithms in observational health settings. *arXiv preprint*, page arXiv:1805.12298, 2018.

[5] V. Feinberg, A. Wan, I. Stoica, M. I. Jordan, J. Gonzalez, and S. Levine. Model-based value estimation for efficient model-free reinforcement learning. In *arXiv preprint arXiv:1803.00101*, 2018.

[6] J. Buckman, D. Hafner, G. Tucker, E. Brevdo, and H. Lee. Sample-efficient reinforcement learning with stochastic ensemble value expansion. In *NeurIPS*, pages 8224–8234, 2018.

[7] L. Buesing, T. Weber, Y. Zwols, S. Racaniere, A. Guez, J. B. Lespiau, and N. Heess. Woulda, coulda, shoulda: Counterfactually-guided policy search. In *arXiv preprint arXiv:1811.06272*, 2018.

[8] Y. Duan, J. Schulman, X. Chen, P. Bartlett, I. Sutskever, and P. Abbeel. Fast reinforcement learning via slow reinforcement learning. In *arXiv preprint arXiv:1611.02779*, 2016.

[9] J. Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick. Learning to reinforcement learn. In *arXiv preprint arXiv:1611.05763*, 2016.

[10] N. Mishra, M. Rohaninejad, X. Chen, and P. Abbeel. A simple neural attentive meta-learner. In *ICLR*, 2018.

[11] K. Rakelly, A. Zhou, D. Quillen, C. Finn, and S. Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *arXiv preprint arXiv:1903.08254*, 2019.

[12] J. Pearl. *Causality: Models, Reasoning, and Inference*. Cambridge University Press, Cambridge, 2000.

[13] S. Shimizu, P.O. Hoyer, A. Hyvärinen, and A.J. Kerminen. A linear non-Gaussian acyclic model for causal discovery. *Journal of Machine Learning Research*, 7:2003–2030, 2006.

[14] P.O. Hoyer, D. Janzing, J. Mooji, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *NIPS*, Vancouver, B.C., Canada, 2009.

[15] K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *UAI*, Montreal, Canada, 2009.

[16] A. Jaiswal, W. AbdAlmageed, Y. Wu, and P. Natarajan. Bidirectional conditional generative adversarial networks. In *arXiv preprint arXiv:1711.07461*, 2017.

[17] M. Deisenroth and C. E. Rasmussen. Pilco: A model-based and data-efficient approach to policy search. In *ICML*, pages 465–472, 2011.

[18] K. Chua, R. Calandra, R. McAllister, and S. Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *NeurIPS*, pages 4754–4765, 2018.

[19] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[20] K. Zhang, Z. Wang, J. Zhang, and B. Schölkopf. On estimation of functional causal models: general results and application to the post-nonlinear causal model. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 7(2):13, 2016.

[21] B. Lang. Monotonic multi-layer perceptron networks as universal approximators. In *International conference on artificial neural networks*, pages 31–37. Springer, 2005.

[22] H. Zhang and Z. Zhang. Feedforward networks with monotone constraints. In *IJCNN'99. International Joint Conference on Neural Networks. Proceedings (Cat. No. 99CH36339)*, volume 3, pages 1820–1823. IEEE, 1999.

[23] M. Oberst and D. Sontag. Counterfactual off-policy evaluation with gumbel-max structural causal models. *arXiv preprint arXiv:1905.05824*, 2019.

[24] A. Raghu, M. Komorowski, I. Ahmed, L. Celi, P. Szolovits, and M. Ghassemi. Deep reinforcement learning for sepsis treatment. In *arXiv preprint arXiv:1711.09602*, 2017.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G.and $\cdots$ Bellemare, and S. Petersen. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[26] H. Hasselt, A. Guez, and D. Silver. Deep reinforcement learning with double Q-learning. In *CoRR*, 2015.

[27] Z. Wang, N. de Freitas, and M. Lanctot. Dueling network architectures for deep reinforcement learning. In *CoRR*, Vancouver, B.C., Canada, 2015.

[28] C. M. Bishop. Mixture density networks. In *Technical Report NCRG/4288, Aston University, Birmingham, UK*, 1994.

[29] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. *arXiv preprint arXiv:1812.02900*, 2018.

[30] A. EW. Johnson, T. J. Pollard, L. Shen, H. L. Li-wei, M. Feng, M. Ghassemi, B. Moody, P. Szolovits, L. A. Celi, and R. G. Mark. Mimic-iii, a freely accessible critical care database. *Scientific data*, 3:160035, 2016.

[31] P. Schulam, F. Wigley, and S. Saria. Clustering longitudinal clinical marker trajectories from electronic health data: Applications to phenotyping and endotype discovery. In *AAAI*, 2015.

[32] S. Schaal. Learning from demonstration. In *NIPS*, pages 1040–1046, 1997.

[33] A. Forney, J. Pearl, and E. Bareinboim. Counterfactual data-fusion for online reinforcement learners. In *Proceedings of the 34th International Conference on Machine Learning*, pages 1156–1164, 2017.

[34] Anirudh Goyal, Philemon Brakel, William Fedus, Soumye Singhal, Timothy Lillicrap, Sergey Levine, Hugo Larochelle, and Yoshua Bengio. Recall traces: Backtracking models for efficient reinforcement learning. *arXiv preprint arXiv:1804.00379*, 2018.

[35] T. Jaakkola, M. I. Jordan, and S. P. Singh. Convergence of stochastic iterative dynamic programming algorithms. In *NIPS*, 1994.

[36] F. S. Melo. Convergence of q-learning: A simple proof. In *http://users.isr.ist.utl.pt/ mtjspaan/readingGroup/ProofQlearning.pdf*, 2016.

A Proof of Theorem 1

Proof. It is clear that as implied by Lemma 1 of [20], the function f and the probabilistic distribution $P(U_{t+1})$ are not uniquely identifiable from the collected data. Furthermore, from the triplets $\langle S_t = s_t, A_t = a, S_{t+1} = s_{t+1}\rangle$, one can always find a infinite number solutions functions to f and $P(U_{t+1})$ such that $S_{t+1} = f(S_t; A_t; U_{t+1})$, where $U_{t+1} \perp\!\!\!\perp (S_t; A_t)$ and f is strictly monotonic in U_{t+1}. Choose an arbitrary solution to f and $P(U_{t+1})$, denoted by f^i and $P^i(U_{t+1})$. Later, surprisingly, we will see that the constructed counterfactual outcome actually does not depend on the index i; that is, it is independent of which f^i and $P^i(U_{t+1})$ we choose.

Given observed evidence $\langle S_t = s_t, A_t = a, S_{t+1} = s_{t+1}\rangle$, because f^i is strictly monotonic in U^i_{t+1}, we can determine $\hat{u}^i_{t+1}$, which is the value of U^i_{t+1}, with

$$\hat{u}^i_{t+1} = {f^i_{s_t,a}}^{-1}(s_{t+1}),$$

where ${f^i_{s_t,a}}^{-1}$ denotes the inverse of f^i with $S_t = s_t, A_t = a$ fixed. Then, we can determine the value of the cumulative distribution function of U^i_{t+1} at $\hat{u}^i_{t+1}$, denoted by α^i.

Without loss of generality, we first show the case where f^i is strictly increasing in U^i_{t+1}. Because f^i is strictly increasing in U^i_{t+1} and $s_{t+1} = f^i(s_t, a, \hat{u}^i_{t+1})$, s_{t+1} is the α^i-quantile of $P(S_{t+1}|S_t = s_t, A_t = a)$. Then it is obvious that since s_{t+1} and $P(S_{t+1}|S_t = s_t, A_t = a)$ are determined, the value of α_i is independent of the index i, that is, it is identifiable. Thus, below, we will use α, instead of α_i.

Since $U_{t+1} \perp\!\!\!\perp (S_t; A_t)$, when doing interventions on A_t, the value $\hat{u}^i_{t+1}$ will not change. The counterfactual outcome $S_{t+1,A_t=a'}|S_t = s_t, A_t = a, S_{t+1} = s_{t+1}$ can be calculated as follows:

$$S_{t+1,A_t=a'} = f^i(S_t = s_t, A_t = a', \hat{u}^i_{t+1}).$$

Because $\hat{u}^i_{t+1}$ does not change after the intervention, the counterfactual outcome $S_{t+1,A_t=a'}|S_t = s_t, A_t = a, S_{t+1} = s_{t+1}$ is the α-quantile of the conditional distribution $P(S_{t+1}|S_t = s_t, A_t = a')$. This quantile exists and it depends only on the conditional distribution $P(S_{t+1}|S_t = s_t, A_t = a')$, but not the chosen function f^i and $P^i(U_{t+1})$, rendering the counterfactual outcome identifiable.

Similarly, the above reasoning procedure can also be applied to the case where f^i is strictly decreasing in U^i_{t+1}.

Therefore, the counterfactual outcome is identifiable, under the condition that f is strictly monotonic in U_{t+1}. □

B Proof of Theorem 2

Proof. From the proof of Theorem 1, we can see that the counterfactual outcome $S_{t+1,A_t=a'}|S_t = s_t, A_t = a, S_{t+1} = s_{t+1}$ is the α-quantile of the conditional distribution $P(S_{t+1}|S_t = s_t, A_t = a')$. Given the transition dynamics $P(S_{t+1}|S_t, A_t)$, the counterfactually augmented triplet can be determined, and it satisfies the underlying Markov decision process (MDP).

Moreover, it has been shown that Q-learning on the data generated from the MDP converges to the optimal value function, under the listed conditions [35, 36]. Therefore, Q-learning on the counterfactually augmented data set converges to the optimal value function Q^*. □

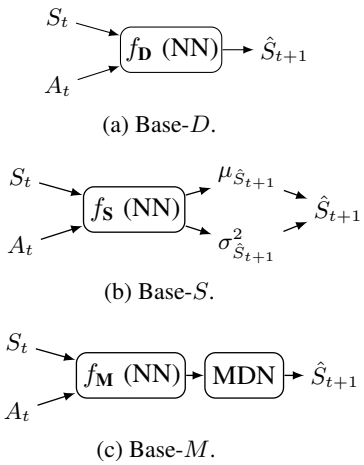
C Three Baselines

Below, we give more details about the three baselines, Base-D, Base-S, and Base-M, which all belong to model-based RL.

Model-based RL explicitly builds a model of the state transition and evaluates actions by searching this model. It is appealing because it is sample efficient. According to the dynamics model that is used, it can be divided into two types: model-based RL with deterministic dynamics models and that with probabilistic ones. Therefore, we compared the proposed CTRL$_g$ and CTRL$_p$ with three well-known baselines in terms of sample efficiency.

Specifically, for model-based RL with a deterministic dynamics model, the next state S_{t+1} is determined by current state S_t and current action A_t: $S_{t+1} = f(S_t, A_t)$, where f can be learned with neural networks. Figure 4(a) gives a graphical illustration of the generating process with a deterministic dynamics model; we denote it by Base-D. It has been observed that model-based RL with deterministic dynamic models tends to converge to sub-optimal solutions where the learned dynamics may not reflect the true process. One way to mitigate this problem is to properly incorporate uncertainty into the dynamics model.

For model-based RL with probabilistic dynamics, one can represent the conditional distribution of the next state given the current state and action in parameterized form, $P(S_{t+1}|S_t, A_t; \theta)$, where θ denotes a set of parameters in the dynamics that needs to be learned. One may parameterize the dynamics model with a Gaussian distribution or a mixture distribution [28] for better generalization. Figure 4(b) illustrates a generating process of mean $\mu_{\hat{S}_{t+1}}$ and variance $\sigma^2_{\hat{S}_{t+1}}$ of a Gaussian distribution, and $\hat{S}_{t+1}$ is sampled from the learned Gaussian distribution; we denote it by Base-S. Figure 4(c), instead, uses a mixture density network (MDN), to handle more general cases; we denote it by Base-M.



(a) Base-D.

(b) Base-S.

(c) Base-M.

Figure 4: Three baselines: (a) Base-D, (b) Base-S, and (c) Base-M.

D More Experimental Results

D.1 More Results on Classical Control Problems

We conduct additional experiments on the data generated under a non-random policy which is a better-than-random policy trained after ten episodes, as shown in Figures 5 and 6.

E Network Architectures

In all experiments, unless stated otherwise, we used linear layers, followed by batch normalization and nonlinear activation function (ReLU) in all network architectures. For simplicity, we omit the notation of batch normalization and ReLU in the following description. Note that, in the implementation of CTRL algorithms, the strict monotonicity stated in **Theorem 1** are easily implemented through monotonic multi-layer perceptron network [21], in which positive signs of the weights are guaranteed by introducing their exponential form [22].

E.1 Network Architecture for Baseline Models

Here we give details about network architectures of the three baseline models. The network structure is presented in Table 1.

Note that, in the model of Base-M, the hyperparameter α, which represents the categorical probability of the mixture density network, is set to 5.

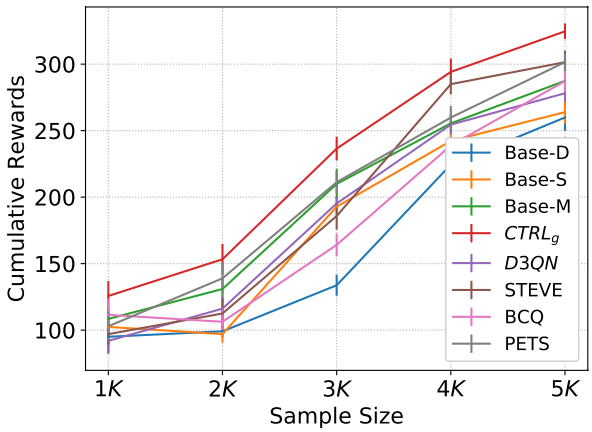


Figure 5: Comparison of CTRL$_g$, CTRL$_p$, three baselines, and three state-of-the-art methods on **SD**.

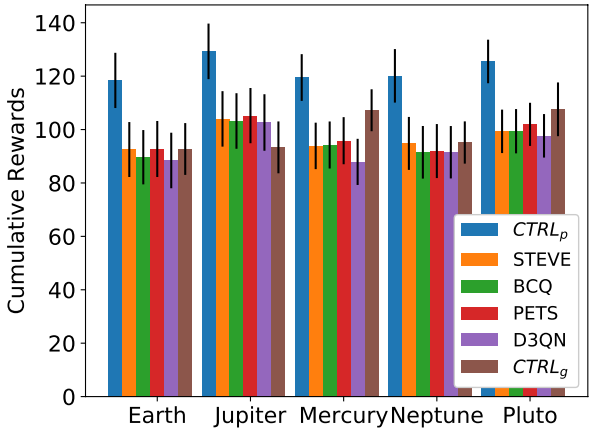


Figure 6: Comparison of CTRL$_g$, CTRL$_p$, and three state-of-the-art methods on **HD**.

Table 1: Network structures of baseline models

	Hidden Layers	Neurons Per Layer
Base-D	2	300
Base-S	2	300
Base-M	2	300

E.2 Network Architecture for CTRL$_g$

Shown in Table 2.

Table 2: Network structures of CTRL$_g$

	Hidden Layers	Neurons Per Layer
Generator	4	$200 \rightarrow 400 \rightarrow 600 \rightarrow 600$
Encoder	4	$600 \rightarrow 600 \rightarrow 400 \rightarrow 200$
Discriminator	4	$600 \rightarrow 600 \rightarrow 400 \rightarrow 200$

E.3 Network Architecture for CTRL$_p$

Shown in Table 3.

Table 3: Network structures of CTRL$_p$

	Hidden Layers	Neurons Per Layer
Generator	4	$200 \rightarrow 400 \rightarrow 600 \rightarrow 600$
Encoder	4	$600 \rightarrow 600 \rightarrow 400 \rightarrow 200$
Discriminator	4	$600 \rightarrow 600 \rightarrow 400 \rightarrow 200$
LSTM	1	200

E.4 Network Architecture for D3QN

Shown in Table 4.

Table 4: Network structures of DDQN

	Hidden Layers	Neurons Per Layer
Policy	4	$512 \rightarrow 512 \rightarrow 512 \rightarrow 512$