DECONFOUNDING REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we propose a general formulation to cope with a family of reinforcement learning tasks in which confounder (i.e., a factor affecting both actions and rewards) exists in dynamic environments. Based on the proposed approach, we extend two representatives of reinforcement learning algorithms: Q-learning and Actor-Critic Methods, to their deconfounding variants. Due to lack of datasets in this direction, a benchmark is developed for deconfounding reinforcement learning algorithms by revising OpenAI Gym and MNIST. We demonstrate that the proposed algorithms are superior to traditional reinforcement learning algorithms in confounding environments. To the best of our knowledge, this is the first time that confounders are taken into consideration for addressing full reinforcement learning problems.

1 INTRODUCTION

In recent years, reinforcement learning has made great progress, spawning a large number of successful applications especially in terms of games (Silver et al., 2016; Mnih et al., 2013; OpenAI, 2018). To the best of our knowledge, however, little work has been done in the direction in which *confounding bias* exists in dynamic environments. Confounding is a causal concept that is described in the language of causality instead of probability and statistics (Pearl, 2009). Confounding bias occurs when a variable influences both who is selected for the treatment and the outcome of the experiment (Pearl & Mackenzie, 2018), which naturally corresponds to the action and the reward in reinforcement learning, respectively. As a matter of fact, confounders have been extensively studied in epidemiology, sociology, and economics. Take for example the widespread kidney stones in which the size of the kidney stone is a confounding factor affecting both the treatment and the recovery (Peters et al., 2017; Pearl, 2009), whether deconfounding the size of the kidney stone or not entirely determines how to choose a more effective treatment. Similarly, in reinforcement learning, if unobserved potential confounders exist, they would affect both actions and rewards when an agent interacts with environments and eventually influence the policy to be optimized.

It is widely acknowledged that one should draw a causal graph before one can achieve any causal conclusion (Pearl, 2009; Pearl & Mackenzie, 2018). Throughout the paper, we assume that, given causal assumptions, we first estimate a model from the observational data we collected from real environments or simulators, and then optimize a policy on the basis of the learned model. This assumption is quite useful in real-world reinforcement learning applications, because in most circumstances, except the data we observed, we either know nothing about real environments or are allowed to do nothing in real environments probably for the sake of ethics, laws or cost. For instance, in healthcare, we can only collect historical medical data such as Electronic Health Records to evaluate the policy rather than directly experiment with patients' lives without evidence that the proposed treatment strategy is better than the current practice (Gottesman et al., 2018); in economics, considering the cost in terms of time and money, it is not practical to study the optimal strategy by actually buying and selling stocks in the market. Hence, in other words, we focus on the observational setting in this paper.

In order to adjust for confounders, we present a general formulation for addressing this class of reinforcement learning problems, namely *deconfounding reinforcement learning*. More specifically, given several common confounding assumptions, we first estimate the corresponding causal model from the observational data, and then deconfound the confounders using the causal language developed by (Pearl, 2009), and finally optimize the policy based on the deconfounding model we calculated. On the basis of the proposed formulation, we extend two popular reinforcement learning

algorithms: Q-learning and Actor-Critic methods to their corresponding deconfounding variants. Due to lack of datasets in this respect, we revise the classic control toolkit in OpenAI Gym (Brockman et al., 2016), making it a benchmark for comparison of deconfounding reinforcement learning algorithms. In addition, we also devise a confounding version of the MNIST dataset (LeCun et al., 1998) to verify the performance of our causal model. Finally, we conduct extensive experiments to demonstrate the superiority of the proposed formulation in deconfounding environments, in comparison to traditional reinforcement learning algorithms.

To sum up, our contributions in this paper are as follows:

1. We propose a general formulation to address a family of reinforcement learning problems in confounding environments, namely *deconfounding reinforcement learning*;
2. We present deconfounding variants of two popular reinforcement learning algorithms: deconfounding Q-learning and deconfounding Actor-Critic methods;
3. We develop a benchmark for deconfounding reinforcement learning by revising the toolkit for classic control in OpenAI Gym (Brockman et al., 2016) and by devising a confounding version of the MNIST dataset (LeCun et al., 1998);
4. We perform a comprehensive comparison of deconfounding reinforcement learning algorithms with their vanilla versions, showing that the proposed approach has an advantage in confounding environments.
5. To the best of our knowledge, this is the first time that we are attempting to build a bridge between confounding and the full reinforcement learning problem. This is one of few research papers aiming at understanding the connections between causal inference and full reinforcement learning.

2 Background

In this section, we briefly review *confounding* in causal inference. We recommend Pearls excellent monograph for further reading (Pearl, 2009; Pearl & Mackenzie, 2018).

2.1 Simpson's Paradox

Let us begin with one of the most famous paradoxes in statistics: Simpson's Paradox. Consider the previously mentioned kidney stones, a classic example of Simpson's paradox (Peters et al., 2017). We collect electronic patient records to investigate the effectiveness of two treatments against kidney stones, where although the overall probability of recovery is higher for patients who took treatment b, treatment a performs better than treatment b on both patients with small kidney stones and with large kidney stones. More precisely, we have

$$\begin{aligned} p(R=1|T=b) &> p(R=1|T=a); \qquad \text{but} \\ p(R=1|T=b, Z=0) &< p(R=1|T=a, Z=0), \\ p(R=1|T=b, Z=1) &< p(R=1|T=a, Z=1); \end{aligned} \tag{1}$$

where Z is the size of the stone, T the treatment, and R the recovery (all binary). How do we cope with this inversion of conclusion? Which treatment do you prefer if you had kidney stones? Does treatment b cause recovery? The answers to these questions depend on the causal relationship between treatment, recovery, and the size of the kidney stone.

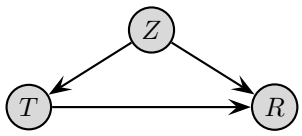


Figure 1: Causal diagram for kidney stones.

2.2 CONFOUNDING

An intuitive explanation for this kidney stone example of Simpson's paradox is that larger stones are more severe than small stones and are much more likely to be treated with treatment a, resulting in that treatment a looks worse than treatment b. Therefore, it is straightforward to assume that the true underlying causal diagram of the kidney stone example is shown in Figure 1, where confounding occurs because the size of kidney stones influences both treatment and recovery. Here, the size of kidney stones is called confounder. The term "confounding" originally meant "mixing" in English, which describes that the true causal effect $T \rightarrow R$ is "mixed" with the spurious correlation between T and R induced by the fork $T \leftarrow Z \rightarrow R$ (Pearl & Mackenzie, 2018). In other words, we will not be able to disentangle the true effect of T on R from the spurious effect if we do not have data on Z. Conversely, if we have measurements of Z, it is easy to deconfound the true and spurious effects by adjusting for Z that averages the effect of T on R in each subgroup of Z (i.e., different size groups in the case of kidney stones).

2.3 DO-OPERATOR AND BACK-DOOR CRITERION

From the viewpoint of causal inference, we can also use the language of intervention, namely *do-operator*, to formulate confounding. In fact, in the example of kidney stones, what we are interested in is how these two treatments compare when we force all patients to take treatment a or b, rather than which treatment has a higher recovery rate given only the observational patient records. Mathematically, we focus on the true effect $p(R = 1|do(T = a))$ (i.e., *intervention distribution* where patients are forced to take treatment a) instead of the spurious effect $p(R = 1|T = a)$ (i.e., *observational distribution* where patients are observed to take treatment a). Therefore, as described previously, confounding can be naturally formulated by the discrepancy between $p(R|T)$ and $p(R|do(T))$.

Generally speaking, do-operator can be calculated in two ways: Randomized Controlled Trials (RCTs) (Fisher, 1935) and Back-Door Criterion (Pearl, 2009). RCTs is the so-called golden standard but rather limited due to many impractical factors (e.g., safety, laws, ethics, physically infeasibility, etc.). Back-door criterion requires a known causal diagram, which applies to our case, in which causal assumptions are provided in advance. According to Back-door criterion, in the kidney stone example, we can immediately attain

$$p(R = 1|do(T = a)) = \sum_{z=0}^{1} p(R = 1|T = a, Z = z)p(Z = z). \tag{2}$$

3 DECONFOUNDING REINFORCEMENT LEARNING

3.1 CAUSAL ASSUMPTIONS

Without loss of generality, as shown in Figure 2, we assume there exists a common confounder in the sequential model, which is time-independent for each individual or for each procedure. This assumption is so general that it would apply to various reinforcement learning tasks across domains. For example, in personalized medicine or precision medicine, socio-economic status can affect both the medication strategy a patient has access to, and the patients general health (Louizos et al., 2017). Therefore socio-economic status acts as confounder between the medication and health outcomes, in which case socio-economic status is time-independent for each patient during the course of treatment. In agriculture, soil fertility may serve as one of confounders affecting both the application of fertilizer and the yield of each plot (Pearl & Mackenzie, 2018). In this circumstance, soil fertility is stable and thought of as a time-independent factor within a period of time (e.g., several months, the growth circle of crops, etc.). In the example of stock markets, apart from socio-economic status as mentioned above, government policy may also act as one of confounders, all of which can be seen time-independent during a reasonable period of time.

3.2 THE MODEL

Given the causal assumption, we first fit a generative model to a sequence of observational data: observations, actions, and rewards, where actions and rewards are confounded by one or several

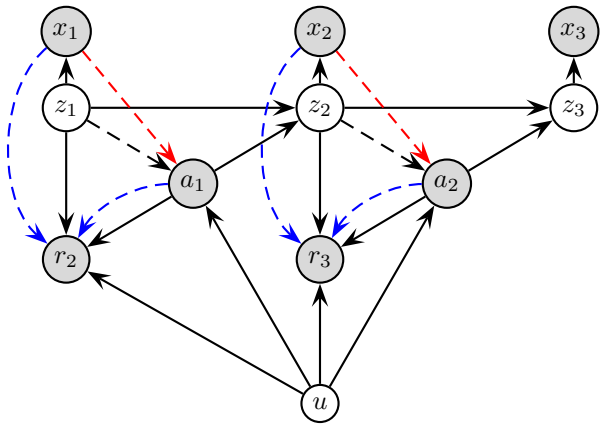


Figure 2: The model for deconfounding reinforcement learning. Solid nodes denote observed variables and open nodes represent unobserved variables. Black dashed lines denote the prior over the possibly true policy learned from the observational data, and red and blue dashed lines denote the variational approximation $q(a_t|x_t)$ and $q(r_{t+1}|x_t, a_t)$, respectively.

unknown factors. Formally, Let $\vec{x} = (x_1, \ldots, x_T)$, $\vec{a} = (a_1, \ldots, a_{T-1})$, $\vec{r} = (r_2, \ldots, r_{T+1})$, $\vec{z} = (z_1, \ldots, z_T)$ be the sequence of observations, actions, rewards, and corresponding latent states, respectively. The confounder is denoted by u, and it is worth noting that here u may stand for more than one confounder in which multiple confounders are seen as a whole represented by u. We assume that $x_t \in \mathbb{R}^{D_x}$, $a_t \in \mathbb{R}^{D_a}$, $r_t \in \mathbb{R}^{D_r}$, $z_t \in \mathbb{R}^{D_z}$, and $u \in \mathbb{R}^{D_u}$, where $D_z \ll D_x$. The generative model for deconfounding reinforcement learning is then given by:

$$p(z_i) = \prod_{i=1}^{D_z} \mathcal{N}(z_{ij}|0,1); \qquad p(u) = \prod_{i=1}^{D_u} \mathcal{N}(z_{ij}|0,1);$$

$$p(x_t|z_t) = \mathcal{N}\left(x_t|\hat{\mu}_t, \hat{\sigma}_t^2\right); \quad \hat{\mu}_t = f_1(z_t) \quad \hat{\sigma}_t^2 = f_2(z_t); \tag{3}$$

$$p(a_t|z_t, u) = \mathcal{N}\left(a_t|\hat{\mu}_t, \hat{\sigma}_t^2\right); \quad \hat{\mu}_t = f_3(z_t, u) \quad \hat{\sigma}_t^2 = f_4(z_t, u); \tag{4}$$

$$p(r_{t+1}|z_t, a_t, u) = \mathcal{N}\left(r_{t+1}|\hat{\mu}_t, \hat{\sigma}_t^2\right); \quad \hat{\mu}_t = f_5(z_t, a_t, u) \quad \hat{\sigma}_t^2 = f_6(z_t, a_t, u); \tag{5}$$

$$p(z_t|z_{t-1}, a_{t-1}) = \mathcal{N}\left(z_t|\hat{\mu}_t, \hat{\sigma}_t^2\right); \quad \hat{\mu}_t = f_7(z_{t-1}, a_{t-1}) \quad \hat{\sigma}_t^2 = f_8(z_{t-1}, a_{t-1}). \tag{6}$$

Note that we parametrize each probability distribution as a Gaussian with its mean and variance modeled by nonlinear functions f_k and each f_k is parametrized by a neural network with its own parameters θ_k for $k = 1, \ldots, 8$. Note that Equation (4) is not necessary in our model, denoted by black dashed lines in Figure 2, but it is potentially useful acting as a prior policy because it is learned from the observational data containing, for example, the real treatment strategies by doctors.

3.3 Learning

Since the nonlinear functions parametrized by neural networks make inference intractable, we will learn the parameters of the model θ_k by employing variational inference along with an inference model, a neural network which approximates the intractable posterior (Rezende et al., 2014; Kingma & Welling, 2013; Krishnan et al., 2015). More specifically, using the variational principle, we posit an approximate posterior distribution $q_\phi(z|x)$ to obtain the following lower bound on the marginal likelihood:

$$\log p_\theta(x) \geq \mathop{\mathbb{E}}_{q_\phi(z|x)} \left[\log p_\theta(x|z)\right] - \mathrm{KL}\left(q_\phi(z|x)||p_\theta(z)\right), \tag{7}$$

where the inequality is by Jensen's inequality and ϕ is the parameter of the inference model $q(z|x)$. Note that in this general case x stands for observational variables and z for latent variables.

3.3.1 VARIATIONAL LOWER BOUND

Directly applying the lower bound in Inequality (7) to our model, we obtain

$$\begin{aligned}\log p_\theta(\vec{x},\vec{a},\vec{r}) &\geq \mathop{\mathbb{E}}_{q_\phi(\vec{z},u|\vec{x},\vec{a},\vec{r})}\left[\log p_\theta(\vec{x},\vec{a},\vec{r}|\vec{z},u)\right] - \mathrm{KL}\left(q_\phi(\vec{z},u|\vec{x},\vec{a},\vec{r})||p_\theta(\vec{z},u)\right)\\ &= \mathcal{L}(\vec{x},\vec{a},\vec{r};\theta,\phi).\end{aligned} \tag{8}$$

Using the Markov property of our model, the full distribution can be factorized in the following way:

$$p_\theta(\vec{x},\vec{a},\vec{r},\vec{z},u) = p(u)p(z_1)\left[\prod_{t=1}^{T} p(x_t|z_t)p(a_t|z_t,u)p(r_{t+1}|z_t,a_t,u)\right]\left[\prod_{t=2}^{T} p(z_t|z_{t-1},a_{t-1})\right]. \tag{9}$$

In addition, for simplicity's sake, we also have the factorization assumption for the posterior approximation:

$$q_\phi(\vec{z},u|\vec{x},\vec{a},\vec{r}) = q(u|\vec{x},\vec{a},\vec{r})q(z_1|\vec{x},\vec{a},\vec{r})\prod_{t=2}^{T} q(z_t|z_{t-1},\vec{x},\vec{a},\vec{r}). \tag{10}$$

Combining Equation (8), (9) and (10) yields:

$$\begin{aligned}\log p_\theta(\vec{x},\vec{a},\vec{r}) \geq& \mathcal{L}(\vec{x},\vec{a},\vec{r};\theta,\phi)\\ =& \sum_{t=1}^{T} \mathop{\mathbb{E}}_{\substack{z_t\sim q(z_t|z_{t-1},\vec{x},\vec{a},\vec{r})\\ u\sim q(u|\vec{x},\vec{a},\vec{r})}}\left[\log p(x_t|z_t) + \log p(a_t|z_t,u) + \log p(r_{t+1}|z_t,a_t,u)\right]\\ &- \mathrm{KL}\left(q(u|\vec{x},\vec{a},\vec{r})||p(u)\right) - \mathrm{KL}\left(q(z_1|\vec{x},\vec{a},\vec{r})||p(z_1)\right)\\ &- \sum_{t=2}^{T} \mathop{\mathbb{E}}_{z_{t-1}\sim q(z_{t-1}|z_{t-2},\vec{x},\vec{a},\vec{r})}\left[\mathrm{KL}\left(q(z_t|z_{t-1},\vec{x},\vec{a},\vec{r})||p(z_t|z_{t-1},a_{t-1})\right)\right],\end{aligned} \tag{11}$$

where we omit subscripts θ and ϕ, and a more detailed derivation can be found in Appendix A. Obviously, Equation (11) is differentiable with respect to the parameters of the model θ and ϕ. Using the reparametrization trick (Kingma & Welling, 2013), we can directly apply backpropagation to update the parameters.

3.3.2 INFERENCE MODEL

From the factorization form in Equation (10), we can see that there are two types of inference models: $q(u|\vec{x},\vec{a},\vec{r})$ and $q(\vec{z}|\vec{x},\vec{a},\vec{r})$. Similar to the generative model in Section 3.2, we also parametrize both of them as Gaussian:

$$q(u|\vec{x},\vec{a},\vec{r}) = \mathcal{N}\left(u|\hat{\mu}_t,\hat{\sigma}_t^2\right); \quad \hat{\mu}_t = f_9(\vec{x},\vec{a},\vec{r}) \quad \hat{\sigma}_t^2 = f_{10}(\vec{x},\vec{a},\vec{r}); \tag{12}$$

$$q(\vec{z}|\vec{x},\vec{a},\vec{r}) = \mathcal{N}\left(\vec{z}|\hat{\mu}_t,\hat{\sigma}_t^2\right); \quad \hat{\mu}_t = f_{11}(\vec{x},\vec{a},\vec{r}) \quad \hat{\sigma}_t^2 = f_{12}(\vec{x},\vec{a},\vec{r}). \tag{13}$$

In fact, as shown in Equation (10), $q(\vec{z}|\vec{x},\vec{a},\vec{r})$ can be further factorized as the product of $q(z_t|z_{t-1},\vec{x},\vec{a},\vec{r})$ for $t = 1,\ldots,T$. Taking a closer look at this term, based on the Markov property of our model, we have $z_t \perp\!\!\!\perp x_1,\ldots,x_{t-1},a_1,\ldots,a_{t-2},r_2,\ldots,r_t|z_{t-1}$, and then the term can be simplified as follows,

$$q(z_t|z_{t-1},\vec{x},\vec{a},\vec{r}) = q(z_t|z_{t-1},a_{t-1},x_t,a_t,r_{t+1},x_{t+1},\ldots,x_T,a_{t+1},\ldots,a_T,r_{t+2},\ldots,r_T). \tag{14}$$

Equation (14) tells us that z_t depends on z_{t-1} and all the current and future observed data $(\vec{x},\vec{a},\vec{r})$. Meanwhile, the conditional independence above means that z_{t-1} contains all the historical data. Therefore, it is natural to calculate z_t based on the whole sequence of data, which is exactly what recurrent neural networks (RNNs) do. Inspired by (Krishnan et al., 2015; 2017), we similarly choose a bi-directional LSTM (Zaremba & Sutskever, 2014) to parameterize f_{11} and f_{12} in Equation (13). Considering Equation (12) has the same structure as Equation (13), f_9 and f_{10} are parameterized by a bi-directional LSTM as well. More details about the architecture can be found in Appendix B.

Note that for the task of sample predictions, that is, at any time step t, given a new x_t, we require to know a_t and r_{t+1} before inferring the distribution over z_t. Hence, we need to introduce two auxiliary distributions, denoted by red and blue dashed lines in Figure 2, to help conduct counterfactual reasoning (i.e., sample prediction on unseen x_t). To be more precise, we have

$$q(a_t|x_t) = \mathcal{N}\left(\mu = \hat{\mu}_t, \sigma^2 = \hat{\sigma}_t^2\right) \quad \hat{\mu}_t = f_{13}(x_t) \quad \hat{\sigma}_t^2 = f_{14}(x_t); \tag{15}$$

$$q(r_{t+1}|x_t, a_t) = \mathcal{N}\left(\mu = \hat{\mu}_t, \sigma^2 = \hat{\sigma}_t^2\right) \quad \hat{\mu}_t = f_{15}(x_t, a_t) \quad \hat{\sigma}_t^2 = f_{16}(x_t, a_t), \tag{16}$$

where f_{13}, f_{14}, f_{15}, and f_{16} are also parameterized by neural networks.

3.4 Deconfounding RL Algorithms

Now we have all the building blocks for deconfounding reinforcement learning algorithms. Once our model is learned from the observational data, it can be directly used as a dynamic environment like those in OpenAI Gym (Brockman et al., 2016). We can exploit the learned model to generate rollouts for policy optimization. In our model, the key difference between traditional and deconfounding reinforcement learning lies in the reward function. To be more precise, assuming that an agent standing at state $z_t = \mathfrak{z}$ performs an action $a_t = \mathfrak{a}$, unlike $p(r_t|z_t = \mathfrak{z}, a_t = \mathfrak{a})$ in traditional reinforcement learning, our deconfounding version based on do-operator as depicted in Section 2.3 is given by

$$p(r_t|z_t = \mathfrak{z}, \mathrm{do}(a_t = \mathfrak{a})) = \int p(r_t|z_t = \mathfrak{z}, a_t = \mathfrak{a}, u)q(u)\mathrm{d}u, \tag{17}$$

where $q(u)$ is the approximate posterior $q(u|\vec{x}, \vec{a}, \vec{r})$ which we compute through the inference network presented in Section 3.3.2. Identification in our case is an immediate result of Pearls back-door criterion mentioned in Section 2.3. Note that the state is fixed while calculating the reward and, therefore, it is unnecessarily taken into account when the back-door criterion is applied. In practice, Equation (17) is approximated using the Monte Carlo method as follows:

$$p(r_t|z_t = \mathfrak{z}, \mathrm{do}(a_t = \mathfrak{a})) = \frac{1}{N}\sum_{i=1}^{N} p(r_t|z_t = \mathfrak{z}, a_t = \mathfrak{a}, u_i) \quad u_i \sim q(u|\vec{x}, \vec{a}, \vec{r}), \tag{18}$$

where N is the number of samples from the approximate posterior of u.

On the basis of our deconfounding reward function, it is straightforward to extend traditional reinforcement learning algorithms to their corresponding deconfounding version. In this paper, we select two representatives of them: Q-learning (Watkins & Dayan, 1992) and Actor-Critic method (Sutton et al., 1998).

Deconfounding Q-Learning Q-learning is one of the most successful approaches in reinforcement learning. Let $Q(z, a; \theta)$ be an approximate action-value function with parameters θ. The parameters θ are learned by iteratively minimizing a sequence of loss functions, where the loss function at step time t is defined as

$$\mathcal{L}_Q(\theta_t) = \mathbb{E}\left(r_{t+1} + \gamma \max_{a_{t+1}} Q(z_{t+1}, a_{t+1}; \theta_{t-1}) - Q(z_t, a_t; \theta_t)\right)^2, \tag{19}$$

where $r_{t+1} \sim p(r_{t+1}|z_t, a_t)$ in vanilla Q-learning whilst $r_{t+1} \sim p(r_{t+1}|z_t, \mathrm{do}(a_t))$ (Equation (18)) in deconfounding Q-learning.

Deconfounding Actor-Critic Methods In contrast to value-based Q-learning, the actor-critic method is a policy-based method directly parameterizing the policy $\pi(a|z; \theta)$, which aims to reduce the variance of the estimate of the policy gradient by subtracting a learned function of the state $b(z)$, known as a baseline, from the return. The learned value function $V(z; \phi)$ is commonly used as the baseline. Taking into consideration that the return is the estimate of $Q(z, a; \phi_Q)$ and $b(z)$ is the estimate of $V(z; \phi_V)$, the gradient of the actor-critic loss function at step time t is given by

$$\nabla J(\theta) = \mathbb{E}_\pi\left[\left(Q(z_t, a_t; \phi_Q) - V(z_t; \phi_V)\right)\nabla_\theta \ln \pi(a_t|z_t; \theta)\right], \tag{20}$$

where $Q(z_t, a_t; \phi_Q) - V(z_t; \phi_V)$ used to be seen as an estimate of the advantage of action a_t in state z_t. In practice, $Q(z_t, a_t; \phi_Q)$ is usually replaced with one-step return, that is, $r_{t+1} + Q(z_{t+1}, a_{t+1}; \phi_Q)$, which brings us back to the same situation described in deconfounding Q-learning. Similarly, the vanilla and deconfounding versions of actor-critic method correspond to $r_{t+1} \sim p(r_{t+1}|z_t, a_t)$ and $r_{t+1} \sim p(r_{t+1}|z_t, \mathrm{do}(a_t))$, respectively.

4 EXPERIMENTAL RESULTS

It is widely acknowledged that evaluating approaches dealing with confounding is always challenging due to lack of groundtruth and benchmark datasets. Besides, little work has been done before in deconfounding reinforcement learning, which renders evaluating such algorithms in this respect much harder. Therefore, we first develop benchmark datasets for evaluation of our algorithms, primarily by revising the MNIST dataset (LeCun et al., 1998) and two environments in OpenAI Gym (Brockman et al., 2016): CartPole and Pendulum. We then evaluate our model as well as compare two proposed deconfounding algorithms to their corresponding vanilla versions on the benchmark datasets we developed.

4.1 IMPLEMENTATION DETAILS

We used Tensorflow (Abadi et al., 2016) for the implementation of our model and deconfounding reinforcement learning algorithms. Optimization was done with Adam (Kingma & Ba, 2014). Unless stated otherwise, the setting of all the hyperparameters and architectures of the neural networks we adopted in this paper can be found in Appendix C.

To verify how good the learned model is, we performed two types of tasks: reconstruction and counterfactual reasoning. The reconstructions were performed by feeding the input sequence into the learned inference network, and then sampling from the resulting posterior distribution according to Equation (13), and finally feeding those samples into the generative network described in Equation (3). The counterfactual reasoning, that is, predicting x_{t+1} given unseen x_t, were executed through four steps: 1) Given an unseen x_t, we estimate a_t and r_{t+1} based on Equation (15) and (16); 2) Once we have x_t, a_t, and r_{t+1}, it is easy to estimate z_t from Equation (13); 3) Using the estimated z_t and a_t, we can directly compute z_{t+1} from Equation (6); (4) The final step is to reconstruct x_{t+1} from z_{t+1} according to Equation (3). Repeating the four steps, we can counterfactually reason out a sequence of data.

To evaluate the confounder u, there are also two scenarios. The easy one is that, given a sequence of observational data $(\vec{x}, \vec{a}, \vec{r})$, it is obvious to estimate u from Equation (12). The more challenging one is to calculate u given only x_t at any time step. Following the same steps used in the task of counterfactual reasoning, we first compute a_t and r_{t+1} based on Equation (15) and (16), and then estimate u through Equation (12).

4.2 CONFOUNDING DATASETS

4.2.1 CONFOUNDING MNIST

Motivated by the Healing MNIST dataset mimicking the healthcare data under harsh conditions (e.g., noisy laboratory measurements, surgeries and drugs affected by patient age and gender, etc.) (Krishnan et al., 2015), we developed a confounding MNIST dataset in which a binary confounder is introduced. More specifically, we select 100 different digits (one hundred 4's and one hundred 5's) to create a synthetic dataset where rotations are encoded as the actions $\vec{a}$ $(-45 \leq a \leq 45)$. First, 20% bit-flip noise are added to each digit image , and then, based on some policy, rotations are performed on each noisy image for five time steps, which produces a large number of 5-step sequences of rotated noisy images. To each generated sequence, exactly one sequence of three consecutive squares (2×2 in pixel) is superimposed with the top-left corner of the images in a random starting location. We treat such generated sequences of images as the observations $\vec{x}$. The training/validation/test set respectively comprises $140000/28000/28000$ sequences of length five. Unless stated otherwise, we keep the same setting in the following two datasets. The definition of reward is as follows. We assume that our goal is to rotate the digit to the canonical view (i.e., the position of 12 o'clock on the clock face, namely upright position) wherever it lies initially. Therefore, the reward is defined as the minus degree between the upright position and the position the digit rotates to. For example, if the digit rotates to the position of 3 o'clock or 9 o'clock, then both rewards are -90.

Now we are arriving at the key stage: how to define the confounder u. Let us first imagine that each sequence of observations represents a sequence of an individual's observed symptoms, the degree of the rotation represents the effectiveness of the treatment, and the reward means to what extent

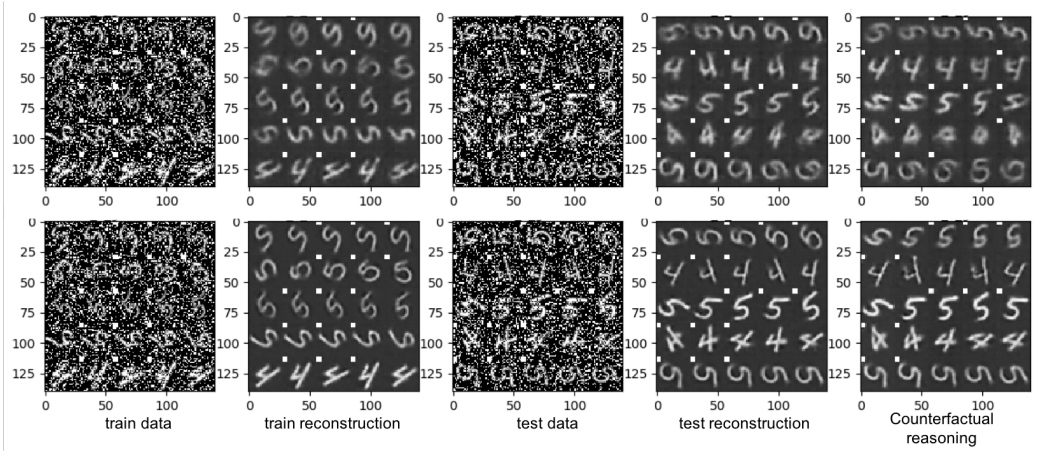


Figure 3: Reconstruction and counterfactual reasoning on the confounding MNIST dataset. Top row: results of a model without confounder; Bottom row: results of our model with confounder as shown in Figure 2.

the patient recovers. Under this circumstance, we assume that the confounder is socio-economic status as mentioned in Section 3.1, and further assume that here the socio-economic status is binary, meaning the rich and the poor, where the rich access more effective treatments $(22.5 \leq |a| \leq 45)$ and recover more quickly (analogy to quicker rotation to the canonical view) whilst the poor receive less effective treatments $(0 \leq |a| < 22.5)$ and recover more slowly. Note that for each sequence, like the unchanged socio-economic status for each patient during the course of treatment, the confounder is fixed, meaning that the range of actions is fixed for generating that sequence.

4.2.2 Confounding Pendulum

To further verify our deconfounding actor-critic method in Section 4.5, we develop a confounding Pendulum dynamic environment by revising the original Pendulum in OpenAI Gym (Brockman et al., 2016). More precisely, we process the screen images of Pendulum in the same way as we do in the confounding MNIST dataset, and treat such generated images as the observations. Similar to Section 4.2.1, we also introduce a binary confounder making the actions divided into two categories $1 \leq |a| \leq 2$ and $0 \leq |a| < 1$. All the others are left unchanged.

4.2.3 Confounding CartPole

Similarly, to verify our deconfounding Q-learning algorithm in Section 4.5, for simplicity, we pay our attention only to the confounding CartPole with discrete actions. We first use the same method as in Section 4.2.2 to deal with screen images. In order to increase the difficulty of the task, we extend the original discrete action space (i.e., $\{0, 1\}$ representing left and right) to a larger space (i.e., $\{-1.0, -0.9, \ldots, 0, 0.1, \ldots, 1.0\}$) with 21 actions. The confounder we imposed partitions the action space into two parts: $\{-0.5, -0.4, \ldots, 0, 0.1, \ldots, 0.5\}$ and the rest. In order to force the confounder to affect the reward, we define the reward as the minus cosine of the pole angle instead of the fixed value of 1 in the original environment.

4.3 Performance analysis of the deconfounding model

Figure 3 presents some samples of reconstruction and counterfactual reasoning on the confounding MNIST dataset. The bottom row is based on our model shown in Figure 2, whilst the top row comes from the same model without the confounder. It is evident that the results generated by our deconfounding model is superior to those produced by the model not taking into account the confounder.

Likewise, some samples of reconstruction and counterfactual reasoning on the confounding Pendulum dataset are shown in Figure 4, where the two models are exactly the same as those used in

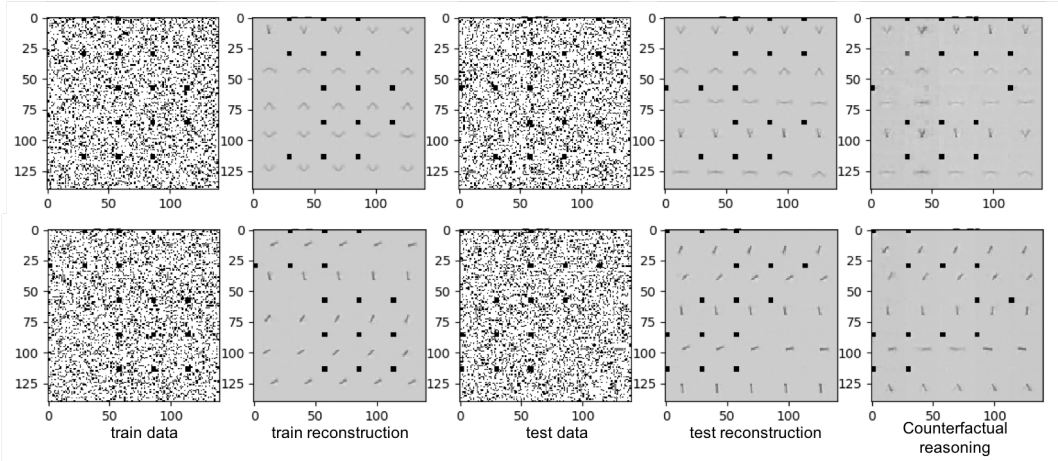


Figure 4: Reconstruction and counterfactual reasoning on the confounding Pendulum dataset. Top row: results of a model without confounder; Bottom row: results of our model with confounder as shown in Figure 2.

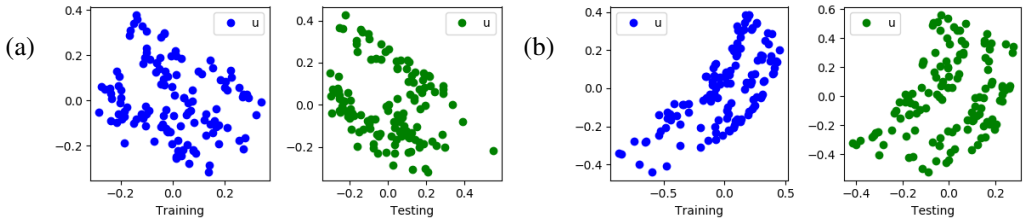


Figure 5: (a) Plot of 128 data points sampling from the posterior approximate of u on the confounding MNIST dataset; (b) Plot of 128 data points sampling from the posterior approximate of u on the confounding CartPole dataset;

Figure 3. In this case, however, it is more straightforward to see the reason why the model without the confounder does not work on the confounding dataset. From the first row, apparently we can see that there are always two pointers appearing in the generated images, because the latent confounder makes the model confusing about the direction the pointer will go. In contrast, such confusion is removed from the second row, in which our deconfounding model can learn the latent confounder and clearly tells the pointer where to go.

4.4 Visualization of the Confounder

As shown in Figure 5, by visualizing the 2-dimensional confounder u, we can discern that although the prior distribution of the confounder is assumed to be a unit Gaussian distribution, the model still can learn two obvious clusters from the data because it is originally a binary variable. It demonstrates that our model has an advantage in learning confounders even if the assumed prior over them were not that accurate.

4.5 Comparison of deconfounding RL algorithms

In this section, we will evaluate the proposed deconfounding Q-learning algorithm and deconfounding Actor-Critic method by comparing them to their vanilla versions. For simplicity's sake, we conduct experiments of Q-learning algorithms on the confounding CartPole dataset which involves discrete actions, while assessing Actor-Critic methods on the confounding Pendulum dataset with continuous actions. In both circumstances, our deconfounding algorithms perform significantly better than their vanilla ones in confounding environments.

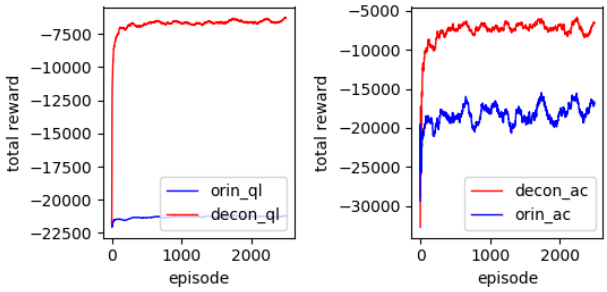


Figure 6: Left: Comparison of original and deconfounding Q-learning algorithms on the confounding CartPole dataset; Right: Comparison of original and deconfounding Actor-Critic methods on the confounding Pendulum dataset.

5 Related Work

Krishnan et al. (2015; 2017) used deep neural networks to model nonlinear state space models and leveraged a structured variational approximation parameterized by recurrent neural networks to mimic the posterior distribution. Levine (2018) reformulated reinforcement learning and control problems to probabilistic inference, which allows us to bear a large pool of approximate inference methods, and flexibly extend the model. Raghu et al. (2017a;b) exploited continuous state-space models and deep reinforcement learning to deduce treatment policies for septic patients from observational data. Gottesman et al. (2018) discussed some issues of evaluating reinforcement learning algorithms in observational health setting. However, all the work mentioned above did not take into account confounders in their models.

Louizos et al. (2017) attempted to learn individual-level causal effects from observational data using variational auto-encoder to estimate the unknown confounder given a causal graph in then non-temporal setting. Paxton et al. (2013) developed predictive models based on electronic medical records without using causal inference. Saria et al. (2010) proposed a nonparametric Bayesian method to analyze clinical temporal data. Soleimani et al. (2017) represented the treatment response curves using linear time-invariant dynamical systems which provides a flexible approach to modeling response over time. Although the latter two work modeled the sequential data, they both do not exploit reinforcement learning or causal inference.

Bareinboim et al. (2015) considered the problem of bandits with unobserved confounders, which is one quite simple reinforcement learning setting without state transitions. Sen et al. (2016) and Ramoly et al. (2017) further studied contextual bandits with latent confounders. Forney et al. (2017) circumvented some problems caused by unobserved confounders in Multi-Armed Bandit by counterfactual-based decision-making. Zhang & Bareinboim (2017) leveraged causal inference to tackle the problem of transferring knowledge across bandit agents. However, all these methods are based on the bandit problem, a simplified version of reinforcement learning, instead of the full reinforcement learning problem.

In fact, as far as we are concerned, this is the first attempt to build a bridge between confounding and the full reinforcement learning problem, and this is also one of few research papers aiming at understanding the connections between causal inference and full reinforcement learning.

6 Conclusion and Future Work

To address the confounding issue in reinforcement learning, we introduced a general formulation, namely deconfounding reinforcement learning. On the basis of the proposed formulation, we presented deconfounding variants of Q-learning and actor-critic methods and showed their superior performance on three confounding datasets that we created by revising OpenAI Gym and MNIST. In the future, we will collaborate with hospitals and apply our approach to real-world medical datasets. We also hope that our work will stimulate further investigation of connections between causal inference and reinforcement learning.

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. Tensorflow: a system for large-scale machine learning. In *OSDI*, volume 16, pp. 265–283, 2016.

Elias Bareinboim, Andrew Forney, and Judea Pearl. Bandits with unobserved confounders: A causal approach. In *Advances in Neural Information Processing Systems*, pp. 1342–1350, 2015.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

Ronald Aylmer Fisher. The design of experiments. 1935.

Andrew Forney, Judea Pearl, and Elias Bareinboim. Counterfactual data-fusion for online reinforcement learners. In *International Conference on Machine Learning*, pp. 1156–1164, 2017.

Omer Gottesman, Fredrik Johansson, Joshua Meier, Jack Dent, Donghun Lee, Srivatsan Srinivasan, Linying Zhang, Yi Ding, David Wihl, Xuefeng Peng, et al. Evaluating reinforcement learning algorithms in observational health settings. *arXiv preprint arXiv:1805.12298*, 2018.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

Rahul G Krishnan, Uri Shalit, and David Sontag. Structured inference networks for nonlinear state space models. In *AAAI*, pp. 2101–2109, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Sergey Levine. Reinforcement learning and control as probabilistic inference: Tutorial and review. *arXiv preprint arXiv:1805.00909*, 2018.

Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. Causal effect inference with deep latent-variable models. In *Advances in Neural Information Processing Systems*, pp. 6446–6456, 2017.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

OpenAI. Openai five. `https://blog.openai.com/openai-five/`, 2018.

Chris Paxton, Alexandru Niculescu-Mizil, and Suchi Saria. Developing predictive models using electronic medical records: challenges and pitfalls. In *AMIA Annual Symposium Proceedings*, volume 2013, pp. 1109. American Medical Informatics Association, 2013.

Judea Pearl. *Causality*. Cambridge university press, 2009.

Judea Pearl and Dana Mackenzie. *The Book of Why*. Allen Lane, 2018.

Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of causal inference: foundations and learning algorithms*. MIT press, 2017.

Aniruddh Raghu, Matthieu Komorowski, Imran Ahmed, Leo Celi, Peter Szolovits, and Marzyeh Ghassemi. Deep reinforcement learning for sepsis treatment. *arXiv preprint arXiv:1711.09602*, 2017a.

Aniruddh Raghu, Matthieu Komorowski, Leo Anthony Celi, Peter Szolovits, and Marzyeh Ghassemi. Continuous state-space models for optimal sepsis treatment-a deep reinforcement learning approach. *arXiv preprint arXiv:1705.08422*, 2017b.

Nathan Ramoly, Amel Bouzeghoub, and Beatrice Finance. A causal multi-armed bandit approach for domestic robots failure avoidance. In *International Conference on Neural Information Processing*, pp. 90–99. Springer, 2017.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

Suchi Saria, Daphne Koller, and Anna Penn. Learning individual and population level traits from clinical temporal data. In *Proceedings of Neural Information Processing Systems*, pp. 1–9. Citeseer, 2010.

Rajat Sen, Karthikeyan Shanmugam, Murat Kocaoglu, Alexandros G Dimakis, and Sanjay Shakkottai. Contextual bandits with latent confounders: An nmf approach. *arXiv preprint arXiv:1606.00119*, 2016.

David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

Hossein Soleimani, Adarsh Subbaswamy, and Suchi Saria. Treatment-response models for counterfactual reasoning with continuous-time, continuous-valued interventions. *arXiv preprint arXiv:1704.02038*, 2017.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press, 1998.

Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992.

Wojciech Zaremba and Ilya Sutskever. Learning to execute. *arXiv preprint arXiv:1410.4615*, 2014.

Junzhe Zhang and Elias Bareinboim. Transfer learning in multi-armed bandit: a causal approach. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 1778–1780. International Foundation for Autonomous Agents and Multiagent Systems, 2017.

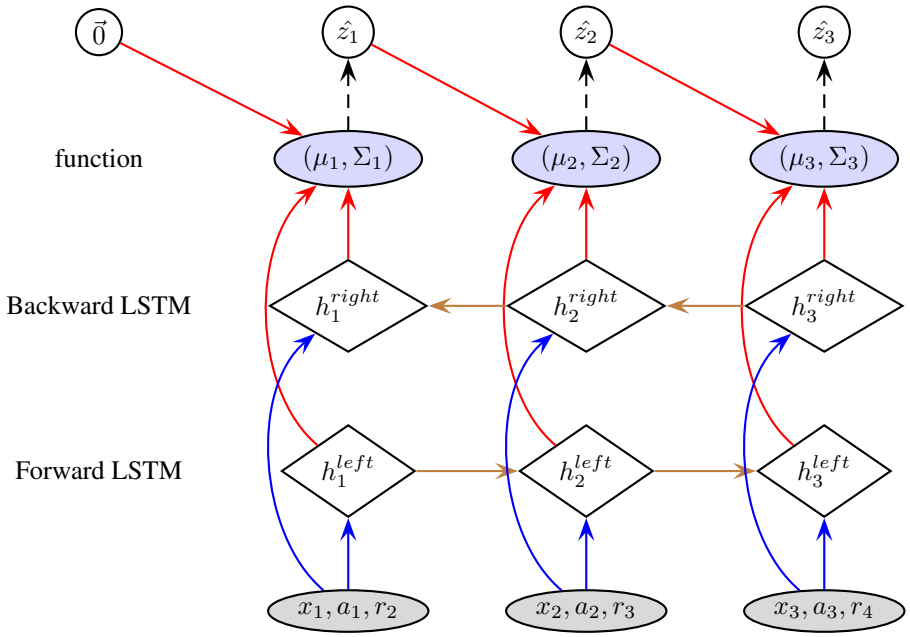
Appendices

A VARIATIONAL LOWER BOUND

$$
\begin{aligned}
&\log p_\theta(\vec{x}, \vec{a}, \vec{r}) \\
=&\log \int_u \int_{\vec{z}} p_\theta(\vec{x}, \vec{a}, \vec{r}, \vec{z}, u) \mathrm{d}\vec{z}\mathrm{d}u \\
\geq& \int_u \int_{\vec{z}} q_\phi(\vec{z}, u|\vec{x}, \vec{a}, \vec{r}) \log \frac{p_\theta(\vec{x}, \vec{a}, \vec{r}, \vec{z}, u)}{q_\phi(\vec{z}, u|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \\
=& \int_u \int_{\vec{z}} q_\psi(u|\vec{x}, \vec{a}, \vec{r}) q_\phi(\vec{z}|\vec{x}, \vec{a}, \vec{r}) \log \frac{p_\theta(\vec{x}, \vec{a}, \vec{r}, \vec{z}, u)}{q_\psi(u|\vec{x}, \vec{a}, \vec{r}) q_\phi(\vec{z}|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \qquad \text{(factorization assumption)} \\
=& \int_u \int_{\vec{z}} q_\psi(u|\vec{x}, \vec{a}, \vec{r}) q_\phi(\vec{z}|\vec{x}, \vec{a}, \vec{r}) \log \frac{p(u)p(z_1)\left[\prod_{t=1}^T p(x_t|z_t)p(a_t|z_t, u)p(r_{t+1}|z_t, a_t, u)\right]\left[\prod_{t=2}^T p(z_t|z_{t-1}, a_{t-1})\right]}{q_\psi(u|\vec{x}, \vec{a}, \vec{r}) q_\phi(\vec{z}|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \\
=& \int_u \int_{\vec{z}} q(u|\vec{x}, \vec{a}, \vec{r}) q(z_1|\vec{x}, \vec{a}, \vec{r}) \cdots q(z_T|z_{T-1}, \vec{x}, \vec{a}, \vec{r}) \\
&\log \frac{p(u)p(z_1)\prod_{t=1}^T p(x_t|z_t)p(a_t|z_t, u)p(r_{t+1}|z_t, a_t, u)\prod_{t=2}^T p(z_t|z_{t-1}, a_{t-1})}{q(u|\vec{x}, \vec{a}, \vec{r}) q(z_1|\vec{x}, \vec{a}, \vec{r}) \cdots q(z_T|z_{T-1}, \vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \\
=& \sum_{t=1}^T \int_u \int_{z_1} \cdots \int_{z_T} q(u|\vec{x}, \vec{a}, \vec{r}) q(z_1|\vec{x}, \vec{a}, \vec{r}) \cdots q(z_T|z_{T-1}, \vec{x}, \vec{a}, \vec{r}) \log \left(p(x_t|z_t)p(a_t|z_t, u)p(r_{t+1}|z_t, a_t, u)\right) \mathrm{d}\vec{z}\mathrm{d}u \\
&+ \int_u \int_{z_1} \cdots \int_{z_T} q(u|\vec{x}, \vec{a}, \vec{r}) q(z_1|\vec{x}, \vec{a}, \vec{r}) \cdots q(z_T|z_{T-1}, \vec{x}, \vec{a}, \vec{r}) \log \frac{p(u)}{q(u|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \\
&+ \int_u \int_{z_1} \cdots \int_{z_T} q(u|\vec{x}, \vec{a}, \vec{r}) q(z_1|\vec{x}, \vec{a}, \vec{r}) \cdots q(z_T|z_{T-1}, \vec{x}, \vec{a}, \vec{r}) \log \frac{p(z_1)}{q(z_1|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \\
&+ \sum_{t=2}^T \int_u \int_{z_1} \cdots \int_{z_T} q(u|\vec{x}, \vec{a}, \vec{r}) q(z_1|\vec{x}, \vec{a}, \vec{r}) \cdots q(z_T|z_{T-1}, \vec{x}, \vec{a}, \vec{r}) \log \frac{p(z_t|z_{t-1}, a_{t-1})}{q(z_t|z_{t-1}, \vec{x}, \vec{a}, \vec{r})} \mathrm{d}\vec{z}\mathrm{d}u \\
=& \sum_{t=1}^T \int_u \int_{z_t} q(u|\vec{x}, \vec{a}, \vec{r}) q(z_t|z_{t-1}, \vec{x}, \vec{a}, \vec{r}) \log \left(p(x_t|z_t)p(a_t|z_t, u)p(r_{t+1}|z_t, a_t, u)\right) \mathrm{d}z_t\mathrm{d}u \\
&+ \int_u q(u|\vec{x}, \vec{a}, \vec{r}) \log \frac{p(u)}{q(u|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}u \\
&+ \int_{z_1} q(z_1|\vec{x}, \vec{a}, \vec{r}) \log \frac{p(z_1)}{q(z_1|\vec{x}, \vec{a}, \vec{r})} \mathrm{d}z_1 \\
&+ \sum_{t=2}^T \int_{z_{t-1}} \int_{z_t} q(z_t|z_{t-1}, \vec{x}, \vec{a}, \vec{r}) \log \frac{p(z_t|z_{t-1}, a_{t-1})}{q(z_t|z_{t-1}, \vec{x}, \vec{a}, \vec{r})} \mathrm{d}z_t\mathrm{d}z_{t-1} \\
=& \sum_{t=1}^T \mathop{\mathbb{E}}_{\substack{z_t \sim q(z_t|z_{t-1}, \vec{x}, \vec{a}, \vec{r}) \\ u \sim q(u|\vec{x}, \vec{a}, \vec{r})}} \left[\log p(x_t|z_t) + \log p(a_t|z_t, u) + \log p(r_{t+1}|z_t, a_t, u)\right] \\
&- \mathrm{KL}\left(q(u|\vec{x}, \vec{a}, \vec{r})||p(u)\right) \\
&- \mathrm{KL}\left(q(z_1|\vec{x}, \vec{a}, \vec{r})||p(z_1)\right) \\
&- \sum_{t=2}^T \mathop{\mathbb{E}}_{z_{t-1} \sim q(z_{t-1}|z_{t-2}, \vec{x}, \vec{a}, \vec{r})} \left[\mathrm{KL}\left(q(z_t|z_{t-1}, \vec{x}, \vec{a}, \vec{r})||p(z_t|z_{t-1}, a_{t-1})\right)\right].
\end{aligned}
$$

B BI-DIRECTIONAL LSTM

In our inference model, we use a similar architecture of bi-directional LSTM to that in (Krishnan et al., 2017). Apart from different inputs, the main difference is to introduce a_{t-1} to computing the combined hidden feature as shown in the following formula.



$$
\begin{aligned}
h_{\text{combined}} &= \frac{1}{4}\left(tanh(W_z \cdot z_{t-1} + b_z) + tanh(W_a \cdot a_{t-1} + b_a) + h_t^{\text{left}} + h_t^{\text{right}}\right) \\
\mu_t &= W_\mu \cdot h_{\text{combined}} + b_\mu \\
\sigma_t^2 &= \text{softplus}\left(W_{\sigma^2} \cdot h_{\text{combined}} + b_{\sigma^2}\right)
\end{aligned}
$$

C EXPERIMENTAL SETTINGS

Hyperparameter	Value
learning rate	0.0001
dimension of z_t	50
dimension of x_t	784
dimension of a_t	1
dimension of r_t	1
dimension of u	2
dimension of LSTM unit	100
batch size	128
number of steps	5
number of epoch	400

Table 1: Hyperparameters of our model for deconfounding reinforcement learning

Function	Architecture
f_1, f_2	$\text{FC}_{512} \rightarrow \text{Conv}_{7,32} \rightarrow \text{Conv}_{14,16} \rightarrow \text{Conv}_{28,1} \rightarrow \text{FC}_{784} \rightarrow \{\text{sigmoid}, \text{softplus}\}$
f_3, f_4	$\{\text{FC}_{100}, \text{FC}_{100}\} \rightarrow \text{FC}_{200} \rightarrow \text{FC}_{200} \rightarrow \text{FC}_1 \rightarrow \{\text{tanh}, \text{softplus}\}$
f_5, f_6	$\{\text{FC}_{100}, \text{FC}_{100}, \text{FC}_{100}\} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_1 \rightarrow \{\text{sigmoid}, \text{softplus}\}$
f_7, f_8	$\{\text{FC}_{100}, \text{FC}_{100}\} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_{50} \rightarrow \{\text{None}, \text{softplus}\}$
f_9, f_{10}	$\{[\text{Conv}_{5,16} \rightarrow \text{Conv}_{5,32} \rightarrow \text{Conv}_{5,32} \rightarrow \text{FC}_{100}], \text{FC}_{100}, \text{FC}_{100}\} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_{100} \rightarrow \text{LSTM}_{100,5} \rightarrow \text{FC}_1 \rightarrow \{\text{None}, \text{softplus}\}$
f_{11}, f_{12}	$\{[\text{Conv}_{5,16} \rightarrow \text{Conv}_{5,32} \rightarrow \text{Conv}_{5,32} \rightarrow \text{FC}_{100}], \text{FC}_{100}, \text{FC}_{100}\} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_{100} \rightarrow \text{LSTM}_{100,5} \rightarrow \text{FC}_{50} \rightarrow \{\text{None}, \text{softplus}\}$
f_{13}, f_{14}	$\text{Conv}_{5,16} \rightarrow \text{Conv}_{5,32} \rightarrow \text{Conv}_{5,32} \rightarrow \text{FC}_1 \rightarrow \{\text{tanh}, \text{softplus}\}$
f_{15}, f_{16}	$\{[\text{Conv}_{5,16} \rightarrow \text{Conv}_{5,32} \rightarrow \text{Conv}_{5,32} \rightarrow \text{FC}_{100}], \text{FC}_{100}\} \rightarrow \text{FC}_{100} \rightarrow \text{FC}_1 \rightarrow \{\text{sigmoid}, \text{softplus}\}$

Table 2: Architectures of our model for deconfounding reinforcement learning. Here FC_k stands for a fully-connected layer with k units, $\text{Conv}_{k,n}$ for a convolution layer with n filters of size $k \times k$, $\text{LSTM}_{n,t}$ for a LSTM layer rolling out for t steps with latent size of n, $\{\cdot\}$ for the parallel operators, and $[\cdot]$ for the sequential operators.