Deconfounding Reinforcement Learning

José Miguel Hernández–Lobato
Department of Engineering
University of Cambridge,
Alan Turing Institute

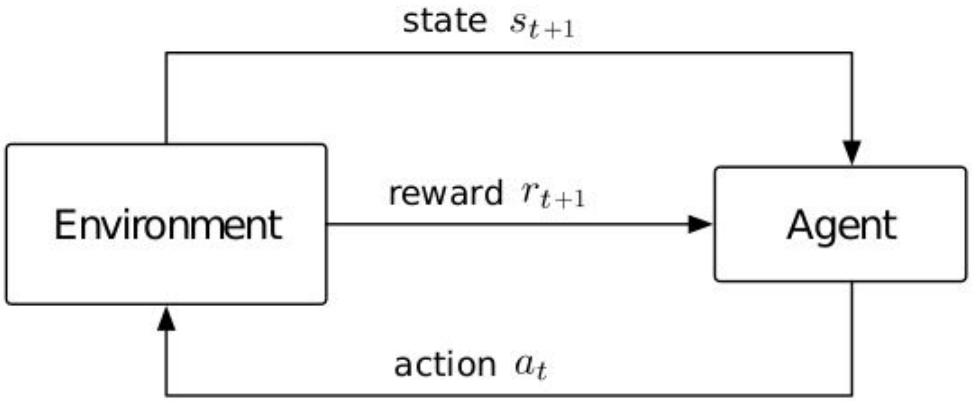
http://jmhl.org, jmh233@cam.ac.uk

Collaborators

Chaochao Lu

Bernhard Schölkopf

The Reinforcement Learning Problem



- state $s_t \in \mathcal{S}$
- action $a_t \in \mathcal{A}$
- reward $r_t \in \mathbb{R}$
- policy $\pi_t(a \mid s) = p(a_t = a \mid s_t = s)$

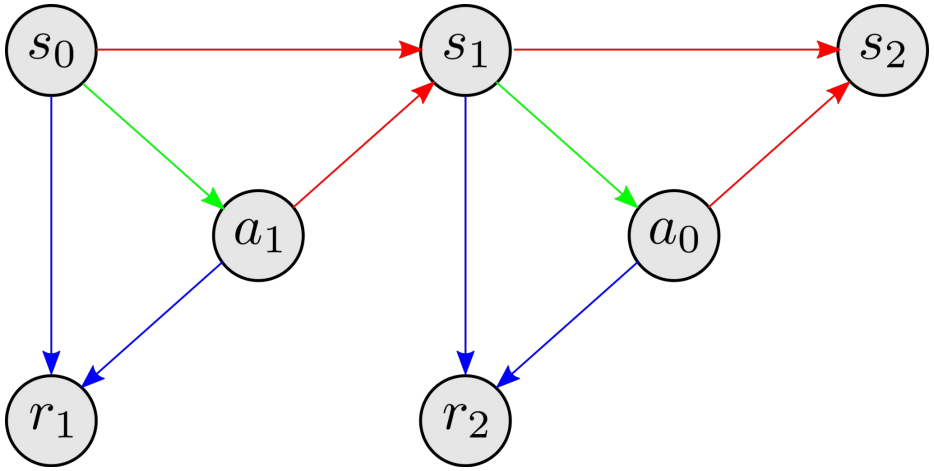
The environment is Markovian:

$$p(s_{t+1}, r_{t+1} | s_0, a_0, r_1, \ldots, s_{t-1}, a_{t-1}, r_t, s_t, a_t) = p(s_{t+1}, r_{t+1} | s_t, a_{t+1}) .$$

The goal is to learn a policy that maximizes the expected sum of discounted rewards:

$$r_1 + \gamma r_2 + \gamma^2 r_3 + \gamma^3 r_4 + \cdots .$$

Graphical Model and Batch Off-policy RL



$p(s_t | s_{t-1}, a_{t-1})$

Environment Dynamics

$p(a_t | s_t)$

Policy

$p(r_t | s_{t-1}, a_{t-1})$

Reward

Batch Off-policy Learning:

To learn the best **target policy** from observed data $D = \{(s_t, a_t, s_{t+1}, r_{t+1})\}$ collected under a different **behaviour policy**, **without collecting new data.**

When working with observational data one has to be careful to avoid **spurious dependencies** between observed variables: **confounding**.

Motivation for a deconfounding approach

Simpson's Paradox

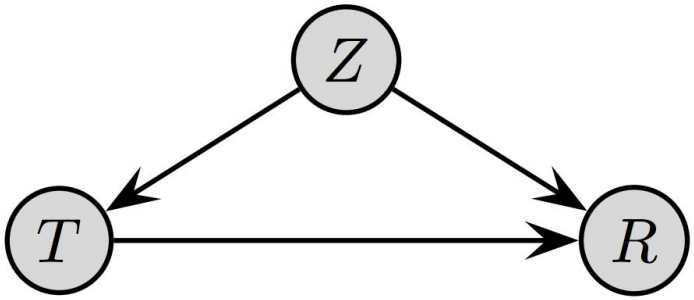
Success rate in treating kidney stones:

	Overall	Patients with small stones	Patients with large stones
Treatment *a*: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment *b*: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

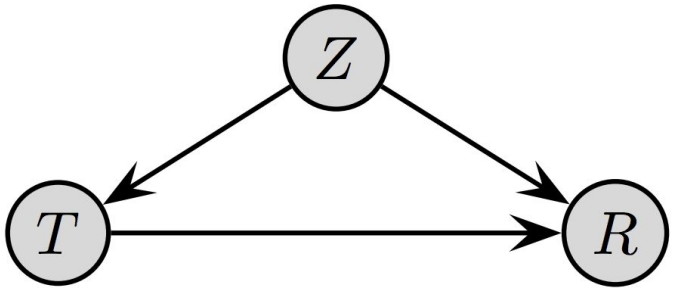
Simpson's Paradox

Success rate in treating kidney stones:

	Overall	Patients with small stones	Patients with large stones
Treatment a: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment b: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

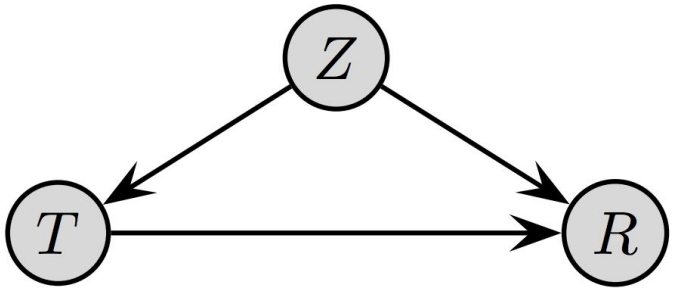


Confounding and Deconfounding

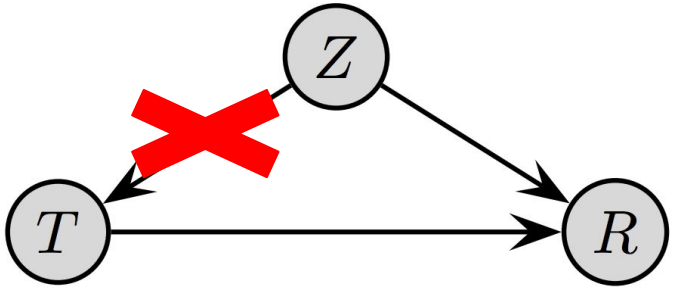


$$P(R = 1|T = 0) \text{ vs } P(R = 1|T = 1)$$

Confounding and Deconfounding

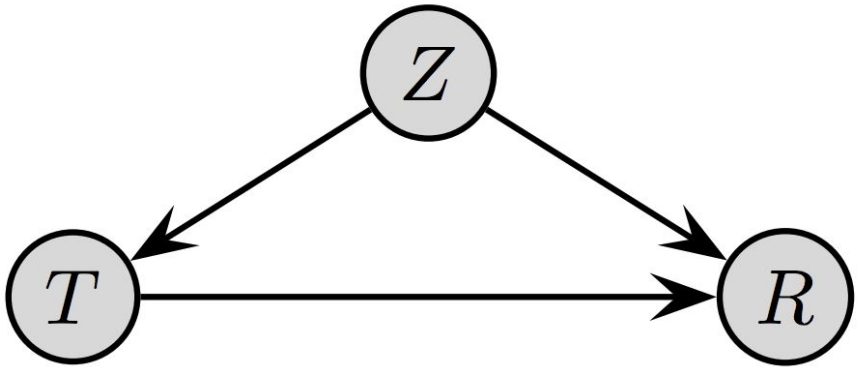


$$P(R = 1|T = 0) \text{ vs } P(R = 1|T = 1)$$



$$P(R = 1|do(T = 0)) \text{ vs } P(R = 1|do(T = 1))$$

Confounding and Deconfounding



$$
\begin{aligned}
& P(R=1|do(T=1)) \\
=& \sum_{Z} P(R=1|T=1,Z)P(Z) \\
=& P(R=1|T=1,Z=0)P(Z=0) + P(R=1|T=1,Z=1)P(Z=1)
\end{aligned}
$$

Simpson's Paradox

	Overall	Patients with small stones	Patients with large stones
Treatment *a*: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment *b*: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

Simpson's Paradox

	Overall	Patients with small stones	Patients with large stones
Treatment *a*: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment *b*: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

$$P(R = 1|T = a)) = 0.78$$
$$P(R = 1|T = b)) = 0.83$$

Simpson's Paradox

	Overall	Patients with small stones	Patients with large stones
Treatment a: Open surgery	78% (273/350)	**93%** (81/87)	**73%** (192/263)
Treatment b: Percutaneous nephrolithotomy	**83%** (289/350)	87% (234/270)	69% (55/80)

$$P(R=1|do(T=a)) \approx 0.93 \cdot \frac{357}{700} + 0.73 \cdot \frac{343}{700} = 0.832.$$

$$P(R=1|do(T=b)) \approx 0.87 \cdot \frac{357}{700} + 0.69 \cdot \frac{343}{700} \approx 0.782.$$

$$P(R=1|T=a)) = 0.78$$

$$P(R=1|T=b)) = 0.83$$

Main Idea

Deconfounding Reinforcement Learning

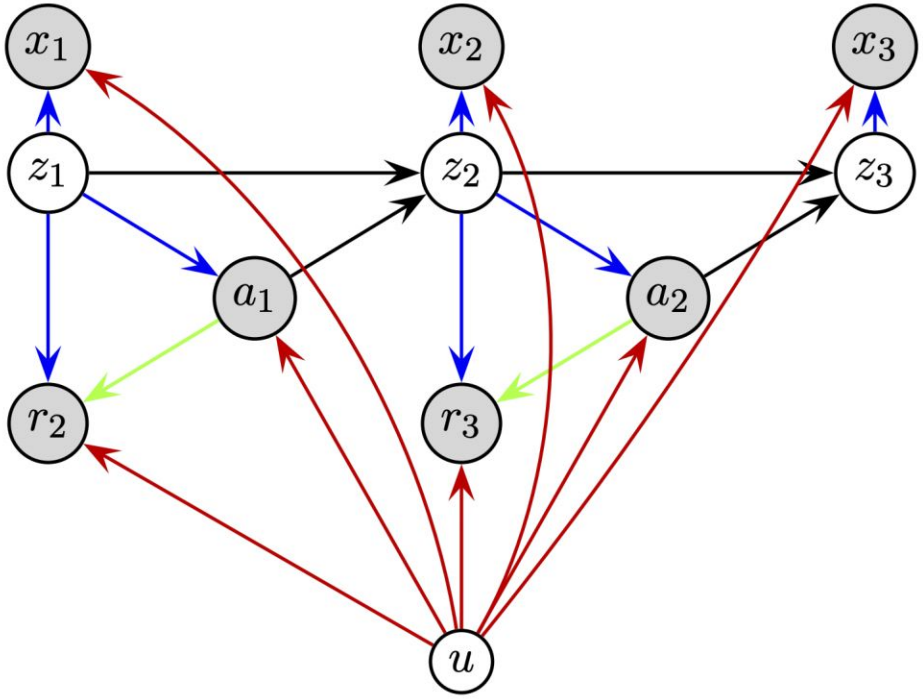
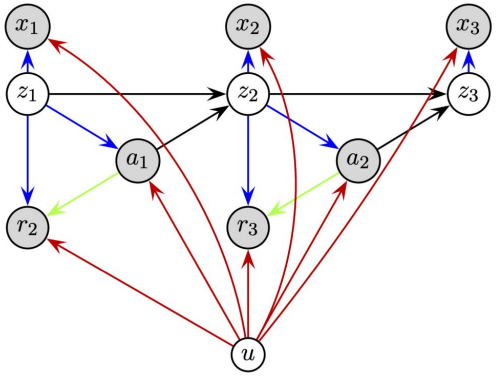


Figure 2: The model for deconfounding reinforcement learning. Grey nodes denote observed variables and white nodes represent unobserved variables. Red and blue arrows emphasize the observed variables affected by u and by z_t, respectively. The causal effects of interest are colored in green.

The Model



$$p(z_t) = \prod_{j=1}^{D_z} \mathcal{N}(z_{tj}|0,1); \qquad p(u) = \prod_{j=1}^{D_u} \mathcal{N}(u_j|0,1);$$

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

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

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

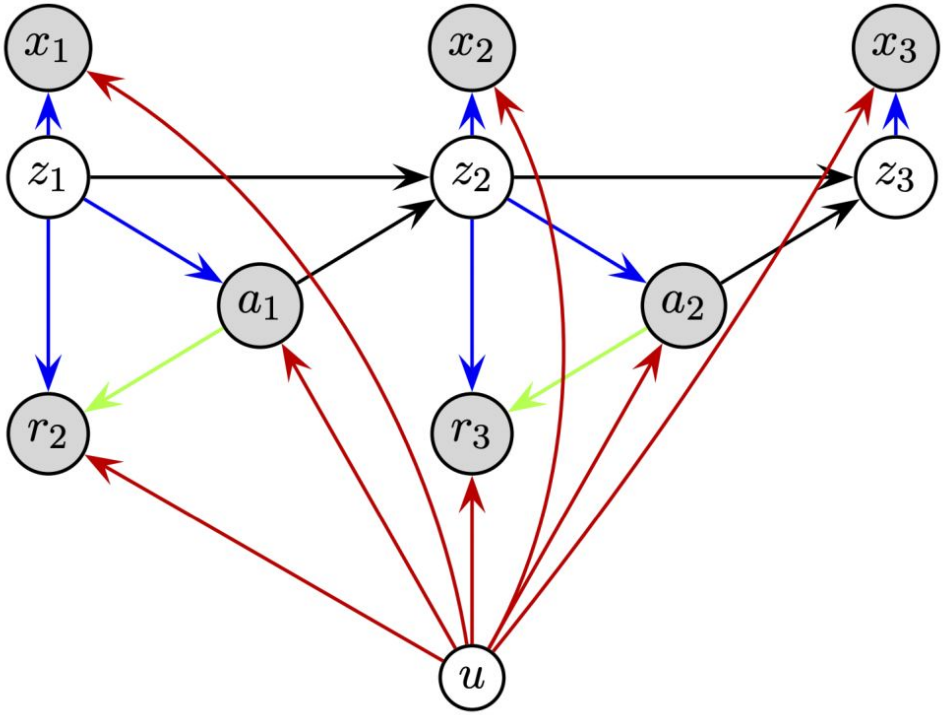
$$p(z_t|z_{t-1},a_{t-1}) = \mathcal{N}\left(z_t|\hat{\mu}_t^z, \hat{\sigma_t^z}^2\right); \quad \hat{\mu}_t^z = f_7(z_{t-1},a_{t-1}), \quad \hat{\sigma_t^z}^2 = f_8(z_{t-1},a_{t-1}); \tag{6}$$

Estimated in practice using **variational inference** with an **inference network** (bidirectional LSTM) that samples the hidden variables given the observed data.

Identification of Causal Effect with Do Calculus

$$p(r_t|z_t, \mathrm{do}(a_t = \mathfrak{a})) = \int_u p(r_t|z_t, \mathrm{do}(a_t = \mathfrak{a}), u)p(u|z_t, \mathrm{do}(a_t = \mathfrak{a}))\mathrm{d}u \tag{7}$$

$$= \int_u p(r_t|z_t, a_t = \mathfrak{a}, u)p(u)\mathrm{d}u, \tag{8}$$



Deconfounding Actor-Critic Methods

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

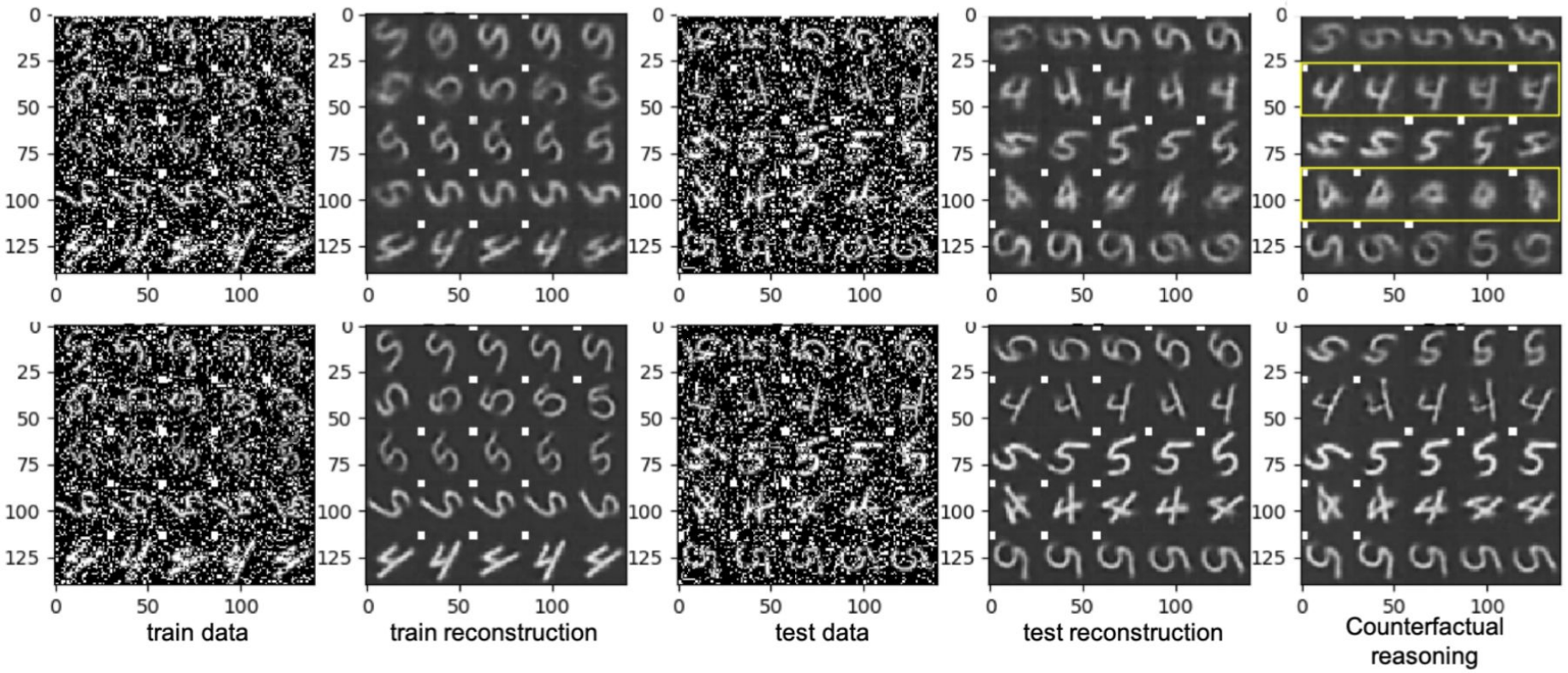
- $Q(z_t, a_t; \phi_Q) - V(z_t; \phi_V)$ is an estimate of the advantage of
- action a_t in state z_t.
- $Q(z_t, a_t; \phi_Q)$ is usually replaced with the one-step return, that is, $r_{t+1} + V(z_{t+1;\phi_V})$.
- The **key difference** in deconfounding actor-critic methods is to use $r_{t+1} \sim p(r_{t+1} | z_t, do(a_t))$.

Experiments

Confounding Datasets: Pendulum and MNIST

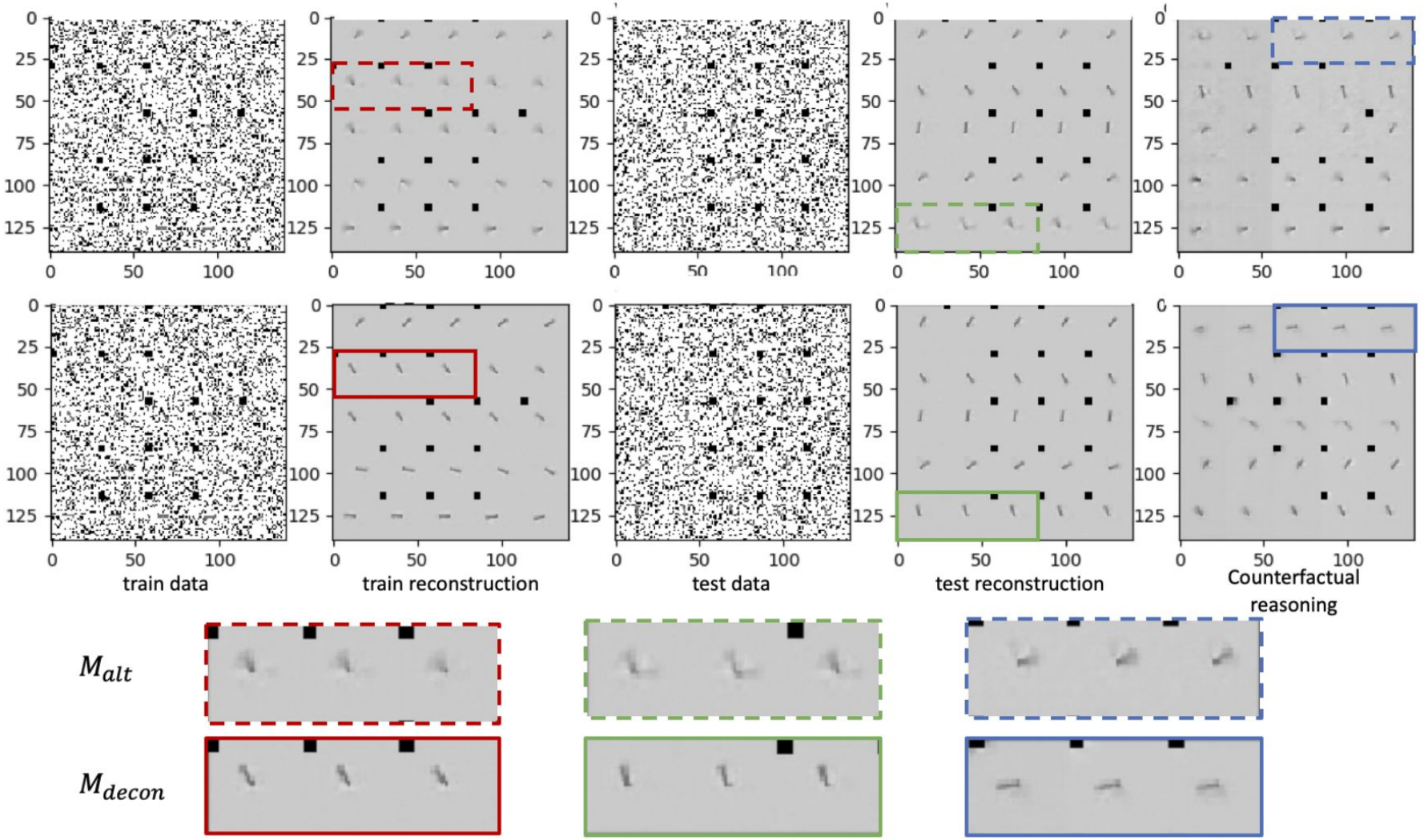
- Select 100 different **screen images** and create a synthetic problem in which actions are joint effort with values between -2 and 2;
- Add 20% **bit-flip noise** to each image.
- Random behaviour policy **confounded** with binary confounder variable u.
- Dataset is formed by a large number of **5-step** sequences of noisy images.
- In each sequence, 3 consecutive images contain a **square block** (2x2 pixels) that is added on top-left corner of the images in a random starting location.
- The **training, validation** and **test** sets with 140K, 28K, and 28K sequences.
- The **reward** is defined as $r = r_o + r_c$, where r_o is the original reward, and $r_c \sim \sum_{r_c} \pi(r_c|a,u)\mathcal{N}(\mu_{r_c}, \sigma^2)$.

Performance Analysis of the Deconfounding Model

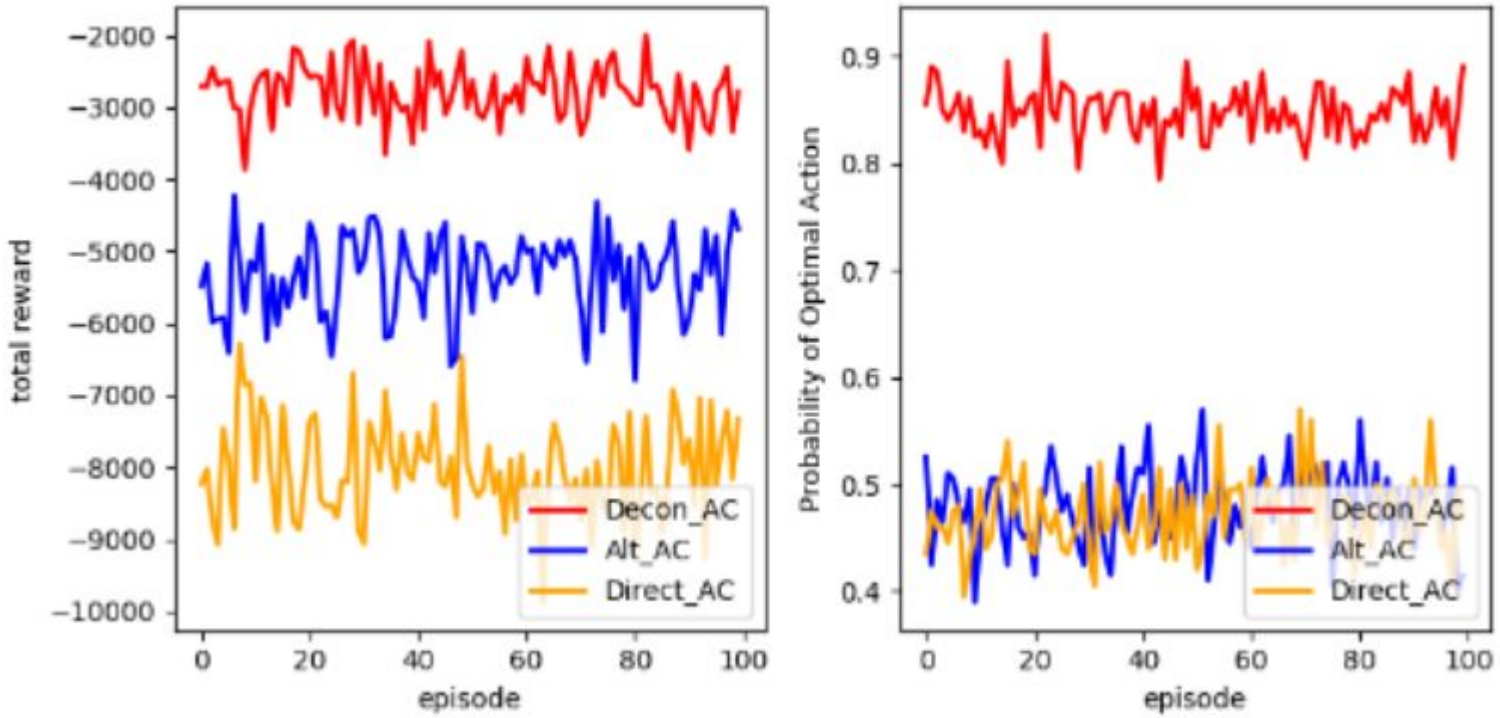


- **Top row:** results from the model **without** the confounder.
- **Bottom row:** results from the model **with** the confounder.
- Sequences boxed in **yellow** have **non-consecutive** white squares and are more blurry.

Performance Analysis of the Deconfounding Model



Comparison of AC Algorithms on Pendulum



Left: total reward over 100 episodes in the testing phase.

Right: probability of optimal action over 100 episodes in the testing phase.

Results on MIMIC-III

Policy	Optimal Trajectories using the Predefined Reward (Avg. CR)	Optimal Trajectories using the Deconfounding Reward (Avg. CR)
Physician	*0.0246*	**0.4147**
Alt_AC	0.0163	0.3211
Direct_AC	0.0153	0.2134
Decon_AC	1.9150	**2.4421**

Take home messages

We focused on **batch off-policy RL** with **unobserved confounders**.

We have...

1. Learnt a **generative model** of the data that includes latent unobserved confounders.

2. Removed the effect of unobserved confounders from the reward, resulting in a **deconfounded reward function**.

3. Used the resulting reward function to extend popular RL methods to their corresponding **deconfounded variant**.

4. Demonstrated gains across several experimental settings.

Thank you.