1 Identifiability

1.1 Tier 1

Lemma 1.1. *Given a countable set X. Let $M(X) = [X, PX]$ the set of stochastic maps from X to X. Note that given an initial distribution, each stochastic map gives a Markov chain. Let $M(X)_{\mathrm{r}} \subset M(X)$ denote the subset of recurrent maps, meaning that for each state $x \in X$, its Markov chain starting with x visits x infinitely often with probability one. Let $\phi_\lambda : M(X) \to P(X^{\mathbb{N}})$ denote the distribution over infinite chains starting with an initial distribution $\lambda \in PX$. Let $C : X^{\mathbb{N}} \to M(X)$ denote the map that gives a maximum likelihood estimate of the transition probabilities given an infinite chain. Let $P(M(X)_{\mathrm{r}})$ denote the set of probabilities over $M(X)_{\mathrm{r}}$. Then for all $m \in M(X)_{\mathrm{r}}, \lambda \in PX$, with probability one, $\tau \sim \phi_\lambda(m)$, $C(\tau) = m$. Furthermore, $\forall p \in P(M(X))_{\mathrm{r}}), \lambda \in PX$, $m \sim p, \tau \sim \phi_\lambda(m)$, $C(\tau)$ is distributed as p.*

Proof. The first result is shown via the ergodic theorem in [Norris, 1997, Theorem 1.10.2]. The second result follows from the fact that $\int_A d\delta_a(a') = a$, where δ_a is the Dirac measure around a (alternatively, the second result follows from a probability monad law). □

Theorem 1.2. *Given countable state and action spaces S, A, let the set $\Theta_\Pi = [S, PA]$ denote the set of stochastic policies, $\Theta_F = [S \times A, PS]$ the set of stochastic dynamics, $\Theta = \Theta_F \times \Theta_\Pi$ the product. Note that each $\theta \in \Theta$ gives a Markov chain with state $S \times A$. Let Θ_{r} be the subset of recurrent Markov chains. Let $p \in P(\Theta_{\mathrm{r}})$ be a joint distribution dynamics and policy giving recurrent chains. Then p can be identified from the distribution over infinite sequences $P((S \times A)^{\mathbb{N}})$ given by rolling out p.*

Proof. Following lemma 1.1, we can identify $p' \in P([S \times A, P(S \times A)])$ that generates the chain. The map $\Theta \to [S \times A, P(S \times A)]$ is injective, so we can recover $p \in P(\Theta_{\mathrm{r}})$ from p'. □

Corollary 1.3. *In Tier 1, assuming the chains are recurrent, we can identify all distributions, thus we can construct the interventional policy.*

1.2 Tier 2

Theorem 1.4. *Given $p(\theta_F, \theta_\Pi) \in P(\Theta)$ and $p'(\theta_F, \theta_\Pi) \in P(\Theta)_{\mathrm{r}}$, such that the marginal distributions over dynamics agree $p(\theta_F) = p'(\theta_F)$. Define on a trajectory τ, $p_{do(A)}(\theta_F|\tau) \propto p(\theta_F) \prod_t p(s_{t+1}|s_t, a_t, \theta_F)$. Assume that with probability one, for policy and dynamics $(\theta_\Pi, \theta_F) \sim p$ and infinite rollout $\tau \sim \phi_\lambda(\theta_\Pi, \theta_F)$, that $p_{do(A)}(\hat{\theta}_F|\tau) = \delta_{\theta_F}$.*

Then, given trajectory distributions $\phi^P_\lambda(p), \phi^P_{\lambda'}(p') \in P((S \times A)^{\mathbb{N}})$, we can identify the interventional policy

$$p(a_t|s_1, do(a_1), ..., s_t) \propto \int_\Theta d\theta_F d\theta_\Pi p(\theta_\Pi, \theta_F) p(a_t|s_t, \theta_\Pi) \prod_t p(s_{t+1}|s_t, a_t, \theta_F)$$

Proof. We identify the dynamics $p(\theta_F)$ using the recurrent model p' via theorem 1.2. Then for a model $p_\psi(a|s, \hat{\theta}_F)$ with parameters ψ, maximise the likelihood of

$$\mathbb{E}_{\theta_F, \theta_\Pi \sim p} \mathbb{E}_{\tau \sim \phi_\lambda(\theta_F, \theta_\Pi)} \mathbb{E}_{\hat{\theta}_F \sim p_{do(A)}(\hat{\theta}_F|\tau)} \log p_\psi(a|s, \hat{\theta}_F)$$

which by assumption is equal to

$$\mathbb{E}_{\theta_F, \theta_\Pi \sim p} \mathbb{E}_{\tau \sim \phi_\lambda(\theta_F, \theta_\Pi)} \log p_\psi(a|s, \theta_F)$$

which is maximised by $p_\psi(a|s, \theta_F) = p(a|s, \theta_F) = \mathbb{E}_{\theta_\Pi \sim p(\theta_\Pi|\theta_F)} p(a|s, \theta_\Pi)$. Then: $p(a_t|s_1, do(a_1), ..., s_t) = \mathbb{E}_{\hat{\theta}_F \sim p_{do(A)}(\hat{\theta}_F|\tau)} p(a|s, \hat{\theta}_F)$. □

2 Optimality

Lemma 2.1. *Given expert distribution $p(\theta_F, \theta_\Pi) \in P(\Theta)$ and agent distribution $p'(\theta_F, \theta_\Pi) \in P(\Theta)$, such that the marginal distributions over dynamics agree $p(\theta_F) = p'(\theta_F)$. Assume the expert's policy is uniquely determined by the environment dynamics (e.g. as it is trained with RL on a single reward function on each environment), so we can write $p(a|s, \theta_F)$ for the expert policy. Assume that with probability one, for policy and dynamics $(\theta_\Pi, \theta_F) \sim p'$ and infinite rollout $\tau \sim \phi_\lambda(\theta_\Pi, \theta_F)$, that $p'_{do(A)}(\hat{\theta}_F|\tau) = \delta_{\theta_F}$.*

Then, on rollouts of p', the interventional policy approaches the expert policy:

$$\lim_{t\to\infty} \mathbb{E}_{\theta_F,\theta_\Pi\sim p'} \mathbb{E}_{\tau\sim\phi_\lambda(\theta_F,\theta_\Pi)} D_{KL}(p(a_t|s_t,\theta_F)||p(a_t|s_1, do(a_1), ..., s_t)) = 0$$

Proof.

$$\begin{aligned}
&\lim_{t\to\infty} \mathbb{E}_{\theta_F,\theta_\Pi\sim p'} \mathbb{E}_{\tau\sim\phi_\lambda(\theta_F,\theta_\Pi)} \log p(a_t|s_t,\theta_F) - \log p(a_t|s_1, do(a_1), ..., s_t) \\
=&\lim_{t\to\infty} \mathbb{E}_{\theta_F,\theta_\Pi\sim p'} \mathbb{E}_{\tau\sim\phi_\lambda(\theta_F,\theta_\Pi)} \log p(a_t|s_t,\theta_F) - \log \mathbb{E}_{\hat{\theta}_F\sim p_{do(A)}(\hat{\theta}_F|\tau_{\le t})} p(a_t|s_t,\hat{\theta}_F) \\
=&\lim_{t\to\infty} \mathbb{E}_{\theta_F,\theta_\Pi\sim p'} \mathbb{E}_{\tau\sim\phi_\lambda(\theta_F,\theta_\Pi)} \log p(a_t|s_t,\theta_F) - \log p(a_t|s_t,\theta_F) = 0
\end{aligned}$$

where in the last line we used that in the limit to infinite sequences, that $p_{do(A)}(\hat{\theta}_F|\tau) = \delta_{\theta_F}$. □

3 Tier 1 algorithm

The paper presents the proof that at tier 1, when suitable demonstrations from the expert are available, the interventional policy is identifiable from the expert data alone, without access to the environment or the expert. In this section, we present a practical algorithm for learning the deconfounded imitation policy from such an expert dataset. At test time, the agent works the same way as the tier 2 algorithm presented in the paper.

Training an inference model directly on the expert demonstration faces the same problem as naive imitation learning, i. e., the trained model takes the expert's actions as evidence for the latent. However, by the assumption stated in appendix 2, that the expert is uniquely determined by the environment dynamics, we can directly learn the conditional policy and dynamics model explaining the expert trajectories from the demonstrations because a latent that explains the dynamics also explains the expert. To train such models, we use variational inference to learn a trajectory encoder $q_{\phi_{\text{off}}}$, which infers the latent for the expert trajectories, and a factorized decoder, which reconstructs the dynamics of the environment and the expert's policy using networks p_ψ and π_η respectively. The variational inference objective is given by

$$\mathcal{L}_{\text{off},i} = \mathbb{E}_{\hat{\theta}\sim q_\phi(\hat{\theta}|\tau_e^i)} \left[\sum_{t=0}^{H} \log p_\psi(s_{t+1} \mid s_t, a_t, \hat{\theta}) + \log \pi_\eta(a_t|s_t,\hat{\theta}) \right] - \beta D_{KL}\left(q_{\phi_{\text{off}}}(\hat{\theta} \mid \tau_e^i) \,\middle\|\, p(\hat{\theta}) \right), \tag{1}$$

which, unlike the objective in the main paper, represents a VAE where the encoder $q_{\phi_{\text{off}}}$ takes as input the full expert trajectory τ_e^i, and the decoder decodes both the action and transition probabilities throughout the trajectory.

This gives us a way for training the conditional policy imitating the experts in the demonstrations and a dynamics model. However, we cannot directly use the learned inference model $q_{\phi_{\text{off}}}$ for implementing the interventional policy, because it takes the expert's actions as evidence for the latent. The tier 2 algorithm works by separately learning an inference model from interactions with the environment and using that inference model for deconfounding the expert trajectories. In tier 1, where we do not have sampling access to the environment, we cannot learn the inference model directly. Instead, we observe that one factor of the decoder used for training the inference model is a dynamics model of the environment conditional on the predicted latent. Therefore, we can use it to generate synthetic trajectories for training an online inference model $q_{\phi_{\text{on}}}$ to minimize the online variational inference objective given in the main paper. The online inference model can then be used for implementing the interventional policy similarly as in tier 2. The full tier 1 algorithm is presented in algorithm 1.

Algorithm 1 Behavior cloning with latent inference, offline variant

Require: The initial parameters of the imitation policy η, offline inference model ϕ_{off}, online inference model ϕ_{on}, dynamics model ψ, prior distribution for the learned latent space $p(\tilde{\theta})$, a dataset of expert trajectories $\{\tau_e^i\}$, an MDP $(\mathcal{S}, \mathcal{A}, p, p_0, H)$, learning rates α_1, α_2, α_3, and α_4.

while not done **do**

- $\tilde{\theta} \sim p(\tilde{\theta})$ ▷ Sample latent from the prior for the learned latent space
- $s_0 \sim p_0(s_0)$ ▷ Sample first state from a learned model or expert data
- $\tau_{\text{synth}} = s_0, t = 0$
- **for** $t \leq H$ **do**
  - $a_t \sim \pi(a_t \mid s_t)$ ▷ Sample action from a Markov policy
  - $s_{t+1} = p_\psi(s_{t+1} \mid s_t, a_t, \tilde{\theta})$ ▷ Dynamics model
  - Append (a_t, s_{t+1}) to τ_{synth}.
  - $t = t + 1$
- **end for**
- $\phi_{\text{on}} = \phi_{\text{on}} - \alpha_1 \nabla_{\phi_{\text{on}}} \hat{\mathcal{L}}(\tau_{\text{synth}})$ ▷ Train the online inference model to minimize equation 6.
- $\phi_{\text{off}} = \phi_{\text{off}} - \alpha_2 \nabla_{\phi_{\text{off}}} \sum_j \hat{\mathcal{L}}_{\text{off}}(\tau_e^j)$ ▷ Train the offline inference model to minimize equation 1.
- $\psi = \psi - \alpha_3 \nabla_\psi \sum_j \hat{\mathcal{L}}_{\text{off}}(\tau_e^j)$ ▷ Train the dynamics model to minimize equation 1.
- $\eta = \eta - \alpha_4 \nabla_\eta \sum_j \hat{\mathcal{L}}_{\text{off}}(\tau_e^j)$ ▷ Train the imitator policy to minimize equation 1.

end while

4 Experimental setup

Implementation details The inference model q_ϕ is implemented as an RNN with GRU architecture Cho et al. [2014] with a hidden layer of 256 units. Before the RNN, the observation is preprocessed by an MLP with two hidden layers of size 256 units and output size 32. The action is preprocessed by a linear transformation to a 32 dimensional vector. The outputs of the RNN are processed by a linear transformation to a vector which parametrizes the latent distribution. The latent distribution is a 256 dimensional Gaussian. One half of the predicted vector represents the mean of the latent distribution and the other half, after softplus activation has been applied to it represents the variance.

The decoder is an MLP with two hidden layers of size 256 and a linear output layer. It uses the same input preprocessing networks for the observations and actions as the inference model.

The policy is an MLP, which takes the latent sample, and an observation as inputs. It uses the same observation embedding network as the other networks and then has two hidden layers with 256 units each.

The naive behavioral cloning baseline uses the same network architecture as the deconfounded algorithm, except it does not represent the belief as a probabilistic latent variable and therefore there is no sampling step. It just directly passes output of the trajectory encoder as the input to the policy network.

All of the MLPs use ReLU activations.

All networks are optimized using the ADAM optimizer [Kingma and Ba, 2014] with otherwise default settings from pytorch [Paszke et al., 2019] apart from the learning rate.

Hyperparameter settings The hyperparameters used for the learning algorithms are presented in table 1.

Computing the ground truth policies All of the probabilities relevant to the bandit problem are known exactly from the definition of the problem and the conditional and interventional policies given in the main paper. Using these probabilities we can compute the true conditional and interventional policies, allowing us to compare the learned algorithms to the relevant optimal policies.

Hyperparameter	Value
Episode length	100
Imitation training steps	5000
Dynamics model training batch size (full episodes)	100
Imitation training batch size (full episodes)	100
Behavioral cloning learning rate	0.001
Variational inference learning rate	0.0001
KL coefficient (β)	0.001

Table 1: Hyperparameters for the deconfounded behavioral cloning and naive behavioral cloning algorithms

In practice, the true belief over theta can be computed for any trajectory as follows

$$\log p(\hat{\theta})_0 = \log \frac{1}{5},\ \log p(\hat{\theta}[A_t])_{t+1} = \begin{cases} \log p(\hat{\theta}[A_t])_t + s_t \log \frac{3}{4} + (1 - s_t) \log \frac{1}{4} & \text{if } A_t = a_t \\ \log p(\hat{\theta}[A_t])_t + s_t \log \frac{1}{4} + (1 - s_t) \log \frac{3}{4} & \text{otherwise} \end{cases}. \tag{2}$$

The true interventional policy can then be computed by sampling a belief $\hat{\theta}_t \sim \log p(\hat{\theta})_t$, and sampling an action from $\pi_{\text{exp}}(a_t|s_t, \hat{\theta}_t)$. This can be seen as a Thompson sampling policy [Thompson, 1933], which acts optimally given its current belief of the task. The true conditional policy is computed similarly, except taking the actions as evidence for the latent is added to the update

$$\log p(\hat{\theta}[A_t])_{t+1} = \begin{cases} \log p(\hat{\theta})_t[A_t] + s_t \log \frac{3}{4} + (1 - s_t) \log \frac{1}{4} + \log \frac{6}{10} & \text{if } A_t = a_t \\ \log p(\hat{\theta})_t[A_t] + s_t \log \frac{1}{4} + (1 - s_t) \log \frac{3}{4} + \log \frac{1}{10} & \text{otherwise} \end{cases}. \tag{3}$$

References

Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *arXiv preprint arXiv:1409.1259*, 2014.

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

J R Norris. *Markov Chains*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1997.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 1933.