Deconfounded Imitation Learning

Risto Vuorio*
Qualcomm AI Research†
University of Oxford
risto.vuorio@cs.ox.ac.uk

Johann Brehmer
Qualcomm AI Research

Hanno Ackermann
Qualcomm AI Research

Daniel Dijkman
Qualcomm AI Research

Taco Cohen
Qualcomm AI Research

Pim de Haan
Qualcomm AI Research
QUVA Lab, University of Amsterdam

Abstract

Standard imitation learning can fail when the expert demonstrators have different sensory inputs than the imitating agent. This is because partial observability gives rise to hidden confounders in the causal graph. We break down the space of confounded imitation learning problems and identify three settings with different data requirements in which the correct imitation policy can be identified. We then introduce an algorithm for deconfounded imitation learning, which trains an inference model jointly with a latent-conditional policy. At test time, the agent alternates between updating its belief over the latent and acting under the belief. We show in theory and practice that this algorithm converges to the correct interventional policy, solves the confounding issue, and can under certain assumptions achieve an asymptotically optimal imitation performance.

1 Introduction

In imitation learning (IL), an agent learns a policy directly from expert demonstrations without requiring the specification of a reward function. This paradigm could be essential for solving real-world problems in autonomous driving and robotics where reward functions can be difficult to shape and online learning may be dangerous. However, standard IL requires that the conditions under which the agent operates exactly match those encountered by the expert. In particular, they assume that there are no *latent confounders*—variables that affect the expert behavior, but that are not observed by the agent. This assumption is often unrealistic. Consider a human driver who is aware of the weather forecast and lowers its speed in icy conditions, even if those are not visible from observations. An imitator agent without access to the weather forecast will not be able to adapt to such conditions.

In such a situation, an imitating agent may take their own past actions as evidence for the values of the confounder. A self-driving car, for instance, could conclude that it is driving fast, thus there can be no ice on the road. This issue of *causal delusion* was first pointed out in Ortega and Braun [2010a,b] and studied in more depth by Ortega et al. [2021]. The authors analyze the causal structure of this problem and argue that an imitator needs to learn a policy that corresponds to a certain interventional distribution. They then show that the classic DAgger algorithm [Ross et al., 2011], which requires querying experts at each time step, solves this problem and converges to the interventional policy.

In this paper, we present a solution to a confounded IL problem, where both the expert policy and the environment dynamics are Markovian. The solution does not require querying experts. We first present a characterization of confounded IL problems depending on properties of the environment

*Work done during internship at Qualcomm AI Research
†Qualcomm AI Research is an initiative of Qualcomm Technologies, Inc.

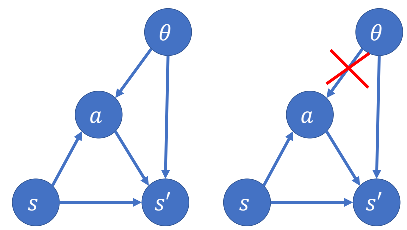
and expert policy (Section 3). We then show theoretically that an imitating agent can learn behaviors that approach optimality when the above Markov assumptions and a recurrence property hold.

We then introduce a practical algorithm for deconfounded imitation learning that does not require expert queries (Section 4). An agent jointly learns an inference network for the value of latent variables that explain the environment dynamics as well as a latent-conditional policy. At test time, the agent iteratively samples latents from its belief, acts in the environment, and updates the belief based on the environment dynamics. An imitator steering a self-driving car, for instance, would learn how to infer the weather condition from the dynamics of the car on the road. This inference model can be applied both to its own online experience as well as to expert trajectories, allowing it to imitate the behavior adequate for the weather.

Finally, our deconfounded imitation learning algorithm is demonstrated in a multi-armed bandit problem. We show that the agent quickly adapts to the unobserved properties of the environment and then behaves optimally (Section 5).

2 Imitation learning and latent confounders

We begin by introducing the problem of confounded imitation learning. Following Ortega et al. [2021], we discuss how behavioral cloning fails in the presence of latent confounders. We then define the interventional policy, which solves the problem of confounded imitation learning.



(a) Expert (b) Ignorant policy

Figure 1: Bayes nets for **(a)** a confounded transition and **(b)** non-confounded transition.

2.1 Imitation learning

Imitation learning learns a policy from a dataset of expert demonstrations via supervised learning. The expert is a policy that acts in a (reward-free) Markov decision process (MDP) defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P(s' \mid s, a), P(s_0))$, where $\mathcal{S}$ is the set of states, $\mathcal{A}$ is the set of actions, $P(s' \mid s, a)$ is the transition probability, and $P(s_0)$ is a distribution over initial states. The expert's interaction with the environment produces a trajectory $\tau = (s_0, a_0, \ldots, a_{T-1}, s_T)$. The expert may maximize the expectation over some reward function, but this is not necessary (and some tasks cannot be expressed through Markov rewards Abel et al. [2021]). In the simplest form of imitation learning, a behavioral cloning policy $\pi_\eta(a \mid s)$ parametrized by η is learned by minimizing the loss $-\sum_{s,a\in\mathcal{D}} \log \pi_\eta(a \mid s)$, where $\mathcal{D}$ is the dataset of state-action pairs collected by the expert's policy.

2.2 Confounded imitation learning

We now extend the imitation learning setup to allow for some variables $\theta \in \Theta$ that are observed by the expert, but not the imitator. We define a family of Markov Decision processes as a latent space Θ, a distribution $P(\theta)$, and for each $\theta \in \Theta$, a reward-free MDP $\mathcal{M}_\theta = (\mathcal{S}, \mathcal{A}, P(s' \mid s, a, \theta), P(s_0 \mid \theta))$. We assume there exists an expert policy $\pi_{\text{exp}}(a \mid s, \theta)$ for each MDP. When it interacts with the environment, it generates the following distribution over trajectories τ:

$$P_{\text{exp}}(\tau \mid \theta) = P(s_0 \mid \theta) \prod_{t=0}^{T} P(s_{t+1} \mid s_t, a_t; \theta) \pi_{\text{exp}}(a_t \mid s_t; \theta)$$

The imitator does not observe the latent θ. It may thus need to implicitly infer it from the past transitions, so we take it to be a non-Markovian policy $\pi_\eta(a_t \mid s_1, a_1, \ldots, s_t)$, parameterized by η. The imitator generates the following distribution over trajectories:

$$P_\eta(\tau \mid \theta) = P(s_0 \mid \theta) \prod_{t=0}^{T} P(s_{t+1} \mid s_t, a_t; \theta)\ \pi_\eta(a_t \mid s_0, a_0, \ldots, s_t)$$

The Bayesian networks associated to these distributions are shown in Figure 1.

The goal of imitation learning in this setting is to learn imitator parameters η such that when the imitator is executed in the environment, the imitator agrees with the expert's decisions, meaning we wish to maximise

$$\mathop{\mathbb{E}}_{\theta\sim P(\theta)} \mathop{\mathbb{E}}_{\tau\sim P_\eta(\tau;\theta)} \left[\sum_{s_t,a_t\in\tau} -\log \pi_{\text{exp}}(a_t \mid s_t, \theta) \right]. \tag{1}$$

If the expert solves some task (e. g. maximizes some reward function), this amounts to solving the same task.

2.3 Naive behavioral cloning

If we have access to a data set of expert demonstrations, one can learn an imitator via behavioral cloning on the expert's demonstrations. At optimality, this learns the *conditional policy*:

$$\pi_{\text{cond}}(a_t \mid s_1, a_1, \dots, s_t) = \mathop{\mathbb{E}}_{\theta\sim p_{\text{cond}}(\theta\mid\tau)} \pi_{\text{exp}}(a_t \mid s_t, \theta) \tag{2}$$

$$p_{\text{cond}}(\theta \mid \tau) \propto p(\theta) \prod_t p(s_{t+1} \mid s_t, a_t, \theta) \pi_{\text{exp}}(a_t \mid s_t, \theta) \tag{3}$$

Following Ortega et al. [2021], consider the following example of a confounded multi-armed bandit with $\mathcal{A} = \Theta = \{1, \dots, 5\}$ and $\mathcal{S} = \{0, 1\}$:

$$p(\theta) = \frac{1}{5}, \quad \pi_{\text{exp}}(a_t \mid s_t, \theta) = \begin{cases} \frac{6}{10} & \text{if } a_t = \theta \\ \frac{1}{10} & \text{if } a_t \neq \theta \end{cases}, \quad P(s_{t+1} = 1 \mid s_t, a_t, \theta) = \begin{cases} \frac{3}{4} & \text{if } a_t = \theta \\ \frac{1}{4} & \text{if } a_t \neq \theta. \end{cases} \tag{4}$$

The expert knows which bandit arm is special (and labeled by θ) and pulls it with high probability, while the imitating agent does not have access to this information.

If we roll out the naive behavioral cloning policy in this environment, shown in Figure 2, we see the causal delusion at work. At time t, the latent that is inferred by p_{cond} takes past actions as evidence for the latent variable. This makes sense on the expert demonstrations, as the expert is cognizant of the latent variable. However, during an imitator roll-out, the past actions are not evidence of the latent, as the imitator is blind to it. Concretely, the imitator will take its first action uniformly and later tends to repeat that action, as it mistakenly takes the first action to be evidence for the latent.

2.4 Interventional policy

A solution to this issue is to only take as evidence the data that was actually informed by the latent, which are the transitions. This defines the following imitator policy:

$$\pi_{\text{int}}(a_t \mid s_1, a_1, \dots, s_t) = \mathop{\mathbb{E}}_{\theta\sim p_{\text{int}}(\theta\mid\tau)} \pi_{\text{exp}}(a_t \mid s_t, \theta), \quad p_{\text{int}}(\theta \mid \tau) \propto p(\theta) \prod_t p(s_{t+1} \mid s_t, a_t, \theta). \tag{5}$$

In a causal framework, that corresponds to treating the choice of past actions as interventions. In the notation of the do-calculus [Pearl, 2009], this equals $p(a_t \mid s_1, \text{do}(a_1), s_2, \text{do}(a_2), \dots, s_t)$. The policy in Equation (5) is therefore known as *interventional policy* [Ortega et al., 2021].

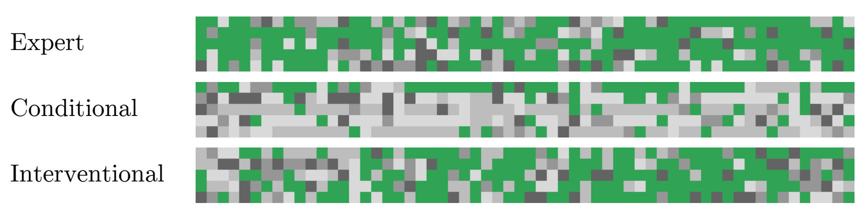


Figure 2: Actions from rollouts from bandit environment (4). The x-axis is episode time. In the y-axis five roll-outs are shown with from the expert and policies (3) and (5). Colors denote actions, with the correct arm labelled green. The interventional imitator tends to the expert policy, while the conditional policy tends to repeat itself.

3 Deconfounding imitation learning

We now present our theoretical results on how imitation learning can be deconfounded. We first show that the interventional policy is optimal in some sense, before analyzing in which settings it can be learned.

3.1 Optimality of the interventional policy

Under some reasonable assumptions, the interventional policy approaches the expert's policy, as we prove in the Appendix B.

Theorem 3.1 (Informal)**.** *If the interventional inference $p_{\text{int}}(\theta \mid \tau_{<t})$ approaches the true latent of the environment as $t \to \infty$ on the rollouts of π_{int}, and if the expert maximises some reward that is fixed across all environments, then as $t \to \infty$, the imitator policy $\pi_{\text{int}}(a_t|s)$ approaches the expert policy.*

Proof. See Lemma 2.1 in the appendix.

The requirement here means that the transition dynamics must be informative about the latent — we consider latent confounders that manifest in the dynamics, not those that affect only the agent behavior.[3] In this case, the interventional policy thus presents a solution to the confounding problem. In the rest of this paper we focus on the question if and how it can be learned from data.

Note that the interventional policy only guarantees *asymptotic* optimal performance. However, it is not guaranteed to be the policy that adapts to a given test environment in the fastest possible way. Efficient exploration may require learning behavior that the expert never demonstrated Zhou et al. [2019].

3.2 Identification

The rest of this section concerns the question: under which assumptions on the model and data availability can we identify the interventional policy? Under the typical rules of do-calculus, identification should be impossible without observing the latent. However, we discern three tiers of decreasing strength of model assumptions and increasing data requirements in which identification is nevertheless possible without observing the latent. In this way, we split the confounding problem discussed by Ortega and Braun [2010a,b], Ortega et al. [2021] into three separate classes, provide conditions for when each class applies, and weaken the requirements for identifiability for two of these classes.

Tier 1: identifiability from demonstrations

In the first tier, we make the strongest assumption and show that the interventional policy is identifiable only from the expert's demonstrations.

Theorem 3.2 (Informal)**.** *Let the expert's demonstrations be recurrent, meaning that each trajectory contains every state and action pair infinitely often. Then the interventional policy is identifiable only from expert demonstrations.*

Proof. See Theorem A.2 in the appendix.

The assumption of recurrence is for instance satisfied if for any state, the expert policy has full support over actions. In that case, on one trajectory, with one value of the latent θ, we are able to accurately estimate the distributions over both the dynamics $p(s' \mid s, a, \theta)$ and the expert $\pi_{\text{exp}}(a \mid s, \theta)$. Then from the distribution over trajectories, we estimate the distribution over distributions and thus identify $p(\theta)$ and the θ-conditional dynamics and expert distributions. The interventional policy can then be constructed from these models via Equation (5).

In practice, this method suffices when the demonstrations are of sufficient length and the expert and dynamics are sufficiently noisy to sufficiently cover the state and action space in each trajectory.

[3]For instance, our results do not apply to latents that specify the task the agent solves but do not affect the environment dynamics at all. That setting, a staple of the meta-RL literature, generally requires a single-shot or few-shot setting where some demonstrations are required for each value of the latent at test time.

Tier 2: identifiability from demonstrations and explorations

In case the expert's demonstrations are not recurrent, but we are able to execute an exploratory policy in the environment, the interventional policy may still be identifiable:

Theorem 3.3 (Informal)**.** *Consider the setting in which in addition to the expert demonstration data, we can collect trajectories by rolling out an exploration policy. Let the exploration data be recurrent. Let the true latents be identifiable from the expert trajectories given the true state-transition model $p(s'|s, a, \theta)$. Then the interventional policy is identifiable from expert demonstrations and exploration data.*

Proof. See Theorem A.4 in the appendix.

How can we identify the interventional policy from expert data and exploration data? Under the assumption of recurrence of exploration trajectories, which is satisfied if the exploration policy has full support and any state is reachable from any state in the environment, the interventional policy can be identified in the following way. First, we estimate the distribution over the state transitions from the exploratory data. This allows us to construct the interventional inference model $p_{\text{int}}(\theta \mid \tau)$ from equation (5). With that model, we can infer the latent for any expert trajectory and learn a policy $\pi_\eta(a \mid s, \theta)$ with behavioral cloning on the demonstrations.

We need to make the crucial assumption that the inference collapses to the true latent of the dynamics — in other words, the state transitions encountered by the expert are informative about the latent variables. (If the expert only visits states where the latent variables do not influence the transition dynamics, it stands to reason that we cannot deconfound the imitator.) Under this assumption, this method identifies the interventional policy, as we prove in Theorem A.4 in the appendix.

We will demonstrate the identifiability of the interventional policy in this tier in practice in the following section.

Tier 3: identifiability from expert queries

If the dynamics of the environment are such that no exploratory policy leads to recurrent trajectories, then the interventional policy can only be identified if we can interact in the environment and query the expert for its decisions. The procedure is to learn a recurrent policy $\pi_\eta(a_t \mid s_1, a_1, \dots, s_t)$, execute it in the enviroment, query the expert — aware of the true latent — for its actions, and train the imitator with maximum likelihood to match those actions. This is similar to the setup in DAgger [Ross et al., 2011]. In Ortega et al. [2021], it is shown that at optimality this converges to the interventional policy.

Examples for problems that fall into tier 3 include cases where the environment dynamics is not Markov given the latent, where the expert policy is not Markov given the latent, or where the expert only visits regions of the state space in which the dynamics are insensitive to the latent.

4 Practical algorithm

We now introduce a practical algorithm for training an agent from expert data in the presence of latent confounders. For simplicity, we focus on tier 2 of the settings discussed in Section 3, i. e. assume access to the expert data as well as the ability to gather more data interactively. In Appendix C, we describe an algorithm that applies to tier 1 (i. e. does not require the ability to gather more data interactively), but faces a more difficult learning problem in practice.

As outlined in Section 3, we use the access to the family of MDPs defined by the latent θ to learn an inference model for the latent variable, infer the latents on the expert trajectories using the learned model, and learn a policy conditioned on the inferred latent, which imitates the expert based on the demonstrations. At test time, the agent uses posterior sampling (or Thompson sampling) [Thompson, 1933]: it alternates between updating its belief about which MDP it is in and acting optimally under its current belief.

Components The agent consists of: (1) an inference model q_ϕ, which maps trajectories $\tau = (s_0, a_0, \dots, s_n)$ to a belief over a latent variable $q_\phi(\hat{\theta} \mid \tau)$; (2) a dynamics model p_ψ, which maps latent, state, and action to a distribution over the next state $p_\psi(s' \mid s, a, \hat{\theta})$; (3) a prior over latents $p(\hat{\theta})$; and (4) a latent-conditional policy $\pi_\eta(a \mid s, \hat{\theta})$.

Training the inference model The inference model q_ϕ, dynamics model p_ψ, and prior $p(\hat{\theta})$ form a latent variable model for trajectory data. Similarly to a variational autoencoder (VAE) [Kingma and Welling, 2013], they are trained by minimizing a variational inference objective[4] given by

$$\mathcal{L}_i = \mathbb{E}_{\hat{\theta} \sim q_\phi(\hat{\theta}|\tau^i_{:t})} \left[\log p_\psi(s_{t+1} \mid s_t, a_t, \hat{\theta}) \right] - \beta D_{KL}\Big(q_\phi(\hat{\theta} \mid \tau^i_{:t}) \Big\| \, p(\hat{\theta}) \Big) . \tag{6}$$

At timestep t, the encoder q_ϕ takes as input the trajectory observed until timestep t, $\tau_{:t} = (s_0, a_0, \ldots, a_{t-1}, s_t)$, and predicts a distribution of the belief over the latent $\hat{\theta}$. The decoder p_ψ is a dynamics model, which predicts the next state s_{t+1} given the state s_t, action a_t, and a sample from the belief distribution. Unlike a VAE, the proposed latent variable model is not an autoencoder, since instead of decoding the input data distribution, the dynamics model p_ψ decodes the future states. To minimize the loss in Equation (6), the inference model needs to encode in the learned latent $\hat{\theta}$ all of the information in the ground-truth latent θ that is important to predict the state transition.

Crucially, the training data for the inference model is collected from from the interactions of an exploration policy with the environment. As exploration policy, we use the latent-conditional policy π_η, sampling $\hat{\theta}$ from the prior $p(\hat{\theta})$. Other exploration policies may be used as long as they explore sufficiently diverse trajectories and do not depend on the true latent θ. However, we cannot simply train the inference model on expert data, as those trajectories where collected from the θ-dependent expert policy. An inference model trained on that data would exhibit the problem of causal delusions that we are trying to solve.

Training the imitation policy The learned inference model is used for inferring the latents in the expert data distribution. These inferences are valid despite the distribution shift from exploratory data to expert data, as the transition model is the same.

Standard imitation learning can then be used to learn a policy conditional on the inferred latents from the expert demonstrations. We use behavioral cloning as the imitation learning method because it is simple and reasonably effective. We show the full training algorithm in algorithm 1.

Test time At test time, the agent faces an environment with an unknown latent and needs to adapt to the correct expert behavior. We solve this problem by alternating between updating a posterior belief over the latent and acting under the current belief, i. e. sampling a latent from the current belief and acting like the corresponding expert.

Concretely, the agent initially samples a latent from the prior $\hat{\theta} \sim p(\hat{\theta})$ and an action $a \sim \pi_\eta(a|s, \hat{\theta})$ to imitate the expert corresponding to that latent. It observes the state transition and computes the posterior belief with the inference network. Another latent is sampled from the updated belief, and so on. Once the inference has converged to match the true latent for the environment, the true expert for the environment will be imitated consistently. We summarize the test-time behavior in algorithm 2.

In practice, using posterior sampling successfully requires that the imitator conditioned on a randomly sampled latent does not end up in states the true expert would not have visited. For example, in a robotic manipulation task, after accidentally pushing the object outside the reach of the robot, none of the expert policies could continue the task. A simple way of avoiding this problem is allowing multiple episodes of interaction with the task, where the environment is reset to a state sampled from the initial state distribution after each episode. Solving this problem for arbitrary environments without resets requires methods beyond behavioral cloning; we leave this for future work.

5 Experiments

To test our method in practice, we conduct an experiment in the multi-armed bandit problem proposed by Ortega et al. [2021] and described in Section 2. With this experiment, we aim to answer the following questions. First, how big is the effect of confounding on naive imitation learning — large enough to justify the use of specialized methods? Second, is our algorithm capable of identifying the interventional policy? Finally, does the interventional policy converge to the expert policy?

[4]Variational inference is not a necessary component of the algorithm, it is just a particular design choice we make for convenience. Another option would be exact inference over a latent variable model, which is in some cases tractable. We have verified empirically that that leads to similar results.

Algorithm 1 Behavioral cloning with latent inference

Require: Initial parameters of the imitation policy η, inference model ϕ, and dynamics model ψ, an expert dataset $\{\tau_e^j\}$, an MDP $\mathcal{M}$, true latent distribution $p(\theta)$, learning rates α_1, α_2, and α_3.
- **while** not done **do**
  - $\theta \sim p(\theta)$
  - $s_0 \sim p_0(s_0)$
  - $\tau = s_0, t = 0$
  - **for** $t \leq H$ **do**
    - $a_t \sim \pi(a_t \mid s_t)$ ▷ Sample action from a Markov policy, e.g. a random policy
    - $s_{t+1} = p(s_{t+1} \mid s_t, a_t, \theta)$ ▷ True dynamics
    - Append (a_t, s_{t+1}) to τ.
    - $t = t + 1$
  - **end for**
  - $\phi = \phi - \alpha_1 \nabla_\phi \hat{\mathcal{L}}(\tau)$ with $\hat{\mathcal{L}}(\tau)$ being a sample estimate of Equation 6.
  - $\psi = \psi - \alpha_2 \nabla_\psi \hat{\mathcal{L}}(\tau)$ with $\hat{\mathcal{L}}(\tau)$ being a sample estimate of Equation 6.
  - Infer latents $\hat{\theta}_H^j \sim q_\phi(\hat{\theta}_H^j \mid \tau_e^j)$ for expert trajectories $\{\tau_e^j\}$.
  - $\eta \leftarrow \eta - \alpha_3 \nabla_\eta \sum_j \sum_{s,a \in \tau_e^j} \log \pi_\eta(a \mid s, \hat{\theta}_H^j)$.
- **end while**

Algorithm 2 Trained agent imitating an expert

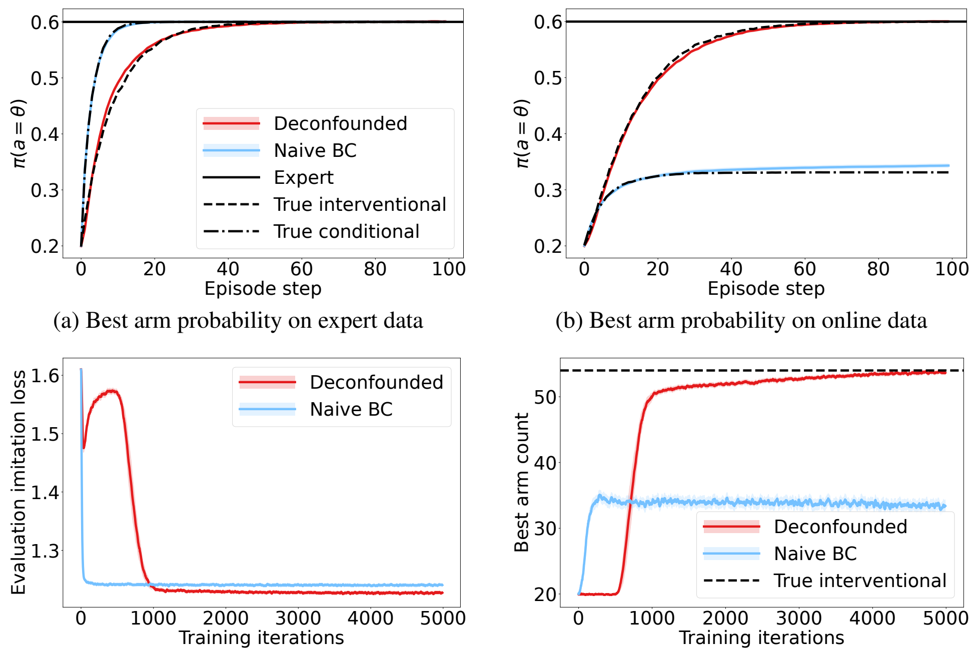
Require: Trained parameters of the imitation policy η, inference model ϕ an MDP $\mathcal{M}$, and a latent θ.
- $s_0 \sim p_0(s_0)$
- $\tau = s_0, t = 0$
- **for** $t \leq H$ **do**
  - $\hat{\theta}_t \sim q_\phi(\hat{\theta}_t \mid \tau)$ ▷ Infer the latent for trajectory
  - $a_t \sim \pi_\eta(a_t \mid s_t, \hat{\theta}_t)$ ▷ Condition on the inferred latent
  - $s_{t+1} = p(s_{t+1} \mid s_t, a_t, \theta)$ ▷ True dynamics
  - Append (a_t, s_{t+1}) to τ.
  - $t = t + 1$
- **end for**

Setup In the experiments, the expert policy is defined in Equation (4). We consider episodes of length 100. We are only interested in the effects of confounding on imitation learning, and not issues arising from optimization. Therefore, in order to avoid overfitting to a finite expert dataset, we resample new data from the expert for each update of the learning algorithms. Each learning algorithm is run for ten independent seeds and the results are averaged. The hyperparameters for the algorithms are provided in Appendix D.

Naive imitation learning and the conditional policy To answer the first question, we compare a naive imitation learner to the true conditional policy described in Section 2. Even the naive learner needs a memory to be able to adapt to a new instance of the bandit defined by the latent. To provide such a memory, we implement the imitation learner as a recurrent neural network (RNN). From Figure 3a, we see that the naive behavioral cloning agent implemented with an RNN learns a policy matching the true conditional policy closely.

This results in problems for the policy learned with naive imitation learning when it is deployed in the environment and has to choose the actions itself, as shown in Figure 3b. In the figure, it can be seen that the naive imitation learner does not converge to the expert policy during the episode. Furthermore, it closely tracks the action probability of the true conditional policy, suggesting that it has suffered the full impact of the confounding problem. Given that the naive imitator closely matches the true conditional policy at convergence, we can conclude that naive imitation learning is unlikely to be enough to solve confounded imitation learning problems.

Deconfounded imitation learning and the interventional policy To answer the second question, we implement the deconfounded imitation learner as described in Section 4. While in principle the inference model can be learned from the data collected by the imitator, we use a random policy for



(a) Best arm probability on expert data

(b) Best arm probability on online data

(c) Imitation loss on expert data during training

(d) Best arm count on online data during training

Figure 3: Experiment with imitation learning in a multi-armed bandit problem. For visual clarity, the curves for the learned agents are smoothed averages for sliding window of length 20. The shading shows the standard error of the mean across random seeds. Panel (a) shows the probability of choosing the best arm on evaluation expert trajectories after training. Panel (b) shows the probability of choosing the best arm in the online environment after training. Panel (c) shows the imitation learning loss on evaluation trajectories over the course of training. Panel (d) shows the number of times the policy chose the best arm during an episode over the course of training.

the data collection. Figures 3a and 3b show that the proposed method (labeled "Deconfounded") closely matches the true interventional policy both on expert trajectories and online. From this we can conclude that in this simple environment, the proposed method produced an inference model and a conditional policy that together match the interventional policy.

Figure 3c shows the imitation loss during training. Notice that the deconfounded algorithm achieves a lower imitation loss than naive BC even though the policy trained with naive BC collapses on the best arm more rapidly on the expert trajectories as shown in Figure 3a. This happens because the deconfounding step in the algorithm conditions the imitator policy on the latent inferred for the whole episode at every step of the episode.

Figure 3d shows the number of times the policies chose the best arm during an episode. This is a convenient proxy for how closely a policy matches the interventional policy. From the figure, it can be seen that the deconfounded imitation learning algorithm closely matches the counterfactual policy, while naive BC converges to lower best arm count.

We conclude that our deconfounded imitation learning algorithm is indeed able to learn the interventional policy, solve the issue of causal delusions faced by naive behavioral cloning, and achieve near-perfect imitation performance. This comes at the price of requiring exploration data to train the inference model as well as an increased number of training iterations needed for convergence.

6 Related work

Imitation learning Imitation learning (or learning from demonstration) has a long history with applications in autonomous driving [Pomerleau, 1988] and robotics [Schaal, 1999]. Standard algorithms include behavioral cloning as well as inverse reinforcement learning Russell [1998], Ng et al. [2000]. Scaling imitation learning to high-dimensional continuous control problems is more challenging and has been solved through adversarial methods [Ho and Ermon, 2016, Fu et al., 2017].

Imitation learning can suffer from a mismatch between the distributions faced by the expert and imitator due to the accumulation of errors when rolling out the imitator policy. This issue is commonly addressed by querying experts during the training [Ross et al., 2011] or by noise insertion in the expert actions. Note that this issue is qualitatively different from the one we discuss in the paper: it is a consequence of the limited support of the expert actions and occurs even in the absence of the latent confounders.

Causality-aware imitation learning De Haan et al. [2019] diagnose the issue of causal confusion in imitation learning, in which the imitator draws causally wrong conclusions from its inputs because such patterns may be easier to learn. This problem is different from the issue of latent confounders we discuss and may occur even if the expert and imitator have access to the same information.

The works most closely related to our paper are Ortega and Braun [2010a,b], Ortega et al. [2021], which point out the issue of latent confounding and causal delusions that we discuss in this paper. In particular, Ortega and Braun [2010b] propose a training algorithm that learns the correct interventional policy. Unlike our algorithm, their approach requires querying experts during the training. However, as we discuss in Section 3, their solution has weaker assumptions and in particular also applies to non-Markovian environment dynamics.

Kumor et al. [2021] study imitation learning on partially observed structural causal models. They characterize when behavioral cloning conditional on observed adjustment variables can maximize the reward. The environments we consider do not meet these criteria, so behavioral cloning fails to imitate the expert, while our method works under additional assumptions. In an extension, Anonymous [2022] propose that an optimal policy can be recovered from a sub-optimal expert whenever the causal effect from the policy on a hypothetical reward would be identifiable from the observables through the IDENTIFY algorithm [Tian and Pearl, 2002], which is not the case for the environments we consider.

Rezende et al. [2020] point out that the same problem appears in partial models, i. e. world models that only use a subset of the (in principle fully observable) state. The part of the state that is not modeled then acts as a confounder for the model predictions. The authors then identify a minimal set of variables that a partial model needs to include in order to not suffer from this confounding issue. While their and our works describe closely related problems, the solutions differ as in our problem we cannot redefine the inputs to include the confounder θ.

Meta-reinforcement learning Our problem is related to meta-reinforcement learning [Duan et al., 2016, Wang et al., 2016]. Here an agent is trained to perform well in a variety of tasks. Both Rakelly et al. [2019] and Zintgraf et al. [2019] propose meta-RL algorithms that consist of a task encoder and a task-conditional policy, similar to our inference model and latent-conditional imitator policy. One difference to our work is that these tasks can vary through the reward function, which is not compatible with imitation learning because the reward function is not available at test time.

Zhou et al. [2019], like our work, consider multiple environments that differ through some latents. They propose agents that learn to probe the environment to determine these latents. The difference to our work is that they work in an RL setting with given reward functions, while our algorithm only requires expert demonstrations.

Another closely related research topic is that of meta-imitation learning [Duan et al., 2017], where the aim is to make imitation learning possible from a small number of demonstrations. Despite both problems centering on imitation learning in a distribution of MDPs, the motivations and methods are different. Our work does not consider adapting to new demonstrations, whereas meta-imitation learning does not consider the confounding problem in the demonstrations.

7 Conclusion

Naive imitation learning algorithms fail in the presence of latent confounders — for instance when the expert has access to more information than the demonstrator. We presented a breakdown of this confounding problem based on when the interventional policy, which solves the confounding issue, is identifiable without query access to the expert. We proposed two algorithms that provably converge to the true interventional policy. In a multi-armed bandit experiment, we demonstrated for one of them that it is able to learn the correct interventional policy, solves the confounding problem that limits naive imitation learning, and converges to the expert behavior.

While our work makes progress theoretically, the empirical demonstration is limited to a simple toy problem. In future work we aim to scale up the experiments to simulated robotics environments. We also plan to adapt the algorithm to other imitation learning algorithms and to study learning objectives that incentivize efficient exploration. A scaled-up algorithm for deconfounded imitation learning may be an important stepping stone on the way to general learning algorithms for control problems, with potentially large impact in the context of robotics and autonomous driving.

References

David Abel, Will Dabney, Anna Harutyunyan, Mark K Ho, Michael Littman, Doina Precup, and Satinder Singh. On the expressivity of markov reward. *Advances in Neural Information Processing Systems*, 34:7799–7812, 2021.

Anonymous. Causal imitation learning via inverse reinforcement learning. https://openreview.net/pdf?id=B-z41MBL_tH, 2022. URL https://openreview.net/pdf?id=B-z41MBL_tH. Accessed: 2022-9-30.

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.

Pim de Haan, Dinesh Jayaraman, and Sergey Levine. Causal confusion in imitation learning. *Adv. Neural Inf. Process. Syst.*, 32, 2019.

Yan Duan, John Schulman, Xi Chen, Peter L Bartlett, Ilya Sutskever, and Pieter Abbeel. Rl^2: Fast reinforcement learning via slow reinforcement learning. *arXiv preprint arXiv:1611.02779*, 2016.

Yan Duan, Marcin Andrychowicz, Bradly C Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. One-Shot imitation learning. March 2017.

Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. October 2017.

Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016.

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.

Daniel Kumor, Junzhe Zhang, and Elias Bareinboim. Sequential causal imitation learning with unobserved confounders. October 2021. URL https://openreview.net/forum?id=o6-k168bBD8.

Andrew Y Ng, Stuart Russell, et al. Algorithms for inverse reinforcement learning. In *ICML*, volume 1, page 2, 2000.

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

PA Ortega and DA Braun. A bayesian rule for adaptive control based on causal interventions. In *Third Conference on Artificial General Intelligence (AGI 2010)*, pages 121–126. Atlantis Press, 2010a.

Pedro A Ortega and Daniel A Braun. A minimum relative entropy principle for learning and acting. *Journal of Artificial Intelligence Research*, 38:475–511, 2010b.

Pedro A Ortega, Markus Kunesch, Grégoire Delétang, Tim Genewein, Jordi Grau-Moya, Joel Veness, Jonas Buchli, Jonas Degrave, Bilal Piot, Julien Perolat, Tom Everitt, Corentin Tallec, Emilio Parisotto, Tom Erez, Yutian Chen, Scott Reed, Marcus Hutter, Nando de Freitas, and Shane Legg. Shaking the foundations: delusions in sequence models for interaction and control. October 2021.

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`.

Judea Pearl. *Causality*. Cambridge University Press, September 2009. URL `https://www.cambridge.org/core/books/causality/B0046844FAE10CBF274D4ACBDAEB5F5B`.

Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Adv. Neural Inf. Process. Syst.*, 1988.

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International conference on machine learning*, pages 5331–5340. PMLR, 2019.

Danilo J Rezende, Ivo Danihelka, George Papamakarios, Nan Rosemary Ke, Ray Jiang, Theophane Weber, Karol Gregor, Hamza Merzic, Fabio Viola, Jane Wang, et al. Causally correct partial models for reinforcement learning. *arXiv preprint arXiv:2002.02836*, 2020.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to No-Regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, Fort Lauderdale, FL, USA, 2011. PMLR.

Stuart Russell. Learning agents for uncertain environments (extended abstract). In *Proceedings of the eleventh annual conference on Computational learning theory - COLT' 98*, New York, New York, USA, 1998. ACM Press.

S Schaal. Is imitation learning the route to humanoid robots? *Trends Cogn. Sci.*, 3(6):233–242, June 1999.

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.

Jin Tian and Judea Pearl. *A general identification condition for causal effects*. eScholarship, University of California, 2002. URL `https://www.aaai.org/Papers/AAAI/2002/AAAI02-085.pdf`.

Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

Wenxuan Zhou, Lerrel Pinto, and Abhinav Gupta. Environment probing interaction policies. *arXiv preprint arXiv:1907.11740*, 2019.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning. *arXiv preprint arXiv:1910.08348*, 2019.

A Identifiability

A.1 Tier 1

Lemma A.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 A.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 A.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 A.3. *In Tier 1, assuming the chains are recurrent, we can identify all distributions, thus we can construct the interventional policy.*

A.2 Tier 2

Theorem A.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 A.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: $p(a_t|s_1, do(a_1), ..., s_t) = \mathbb{E}_{\hat{\theta}_F \sim p_{do(A)}(\hat{\theta}_F|\tau)} p(a|s, \hat{\theta}_F)$. □

B Optimality

Lemma B.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.

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

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

C 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 B, 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

$$\mathcal{L}_{\text{off},i} = \mathbb{E}_{\hat{\theta} \sim q_\phi(\hat{\theta}|\tau_e^i)} \left[\sum_{t=0}^{H} \log p_\psi(s_{t+1} \mid s_t, a_t, \hat{\theta}) + \log \pi_\eta(a_t|s_t, \hat{\theta}) \right] - \beta D_{KL}\left(q_{\phi_{\text{off}}}(\hat{\theta} \mid \tau_e^i) \;\middle\|\; p(\hat{\theta}) \right), \tag{7}$$

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 3.

Algorithm 3 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
    θ̃ ~ p(θ̃)                                ▷ Sample latent from the prior for the learned latent space
    s_0 ~ p_0(s_0)                          ▷ Sample first state from a learned model or expert data
    τ_synth = s_0, t = 0
    for t ≤ H do
        a_t ~ π(a_t | s_t)                   ▷ Sample action from a Markov policy
        s_{t+1} = p_ψ(s_{t+1} | s_t, a_t, θ̃)    ▷ Dynamics model
        Append (a_t, s_{t+1}) to τ_synth.
        t = t + 1
    end for
    φ_on = φ_on − α_1 ∇_{φ_on} L̂(τ_synth)       ▷ Train the online inference model to minimize equation 6.
    φ_off = φ_off − α_2 ∇_{φ_off} Σ_j L̂_off(τ_e^j)  ▷ Train the offline inference model to minimize equation 7.
    ψ = ψ − α_3 ∇_ψ Σ_j L̂_off(τ_e^j)          ▷ Train the dynamics model to minimize equation 7.
    η = η − α_4 ∇_η Σ_j L̂_off(τ_e^j)          ▷ Train the imitator policy to minimize equation 7.
end while
```

D 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

$$\log p(\hat{\theta}_0) = \log \frac{1}{5},\ \log p(\hat{\theta}_{t+1}[A_t]) = \begin{cases} \log p(\hat{\theta}_t[A_t]) + s_t \log \frac{3}{4} + (1-s_t) \log \frac{1}{4} & \text{if } A_t = a_t \\ \log p(\hat{\theta}_t[A_t]) + s_t \log \frac{1}{4} + (1-s_t) \log \frac{3}{4} & \text{otherwise} \end{cases}. \tag{8}$$

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

$$\log p(\hat{\theta}_{t+1}[A_t]) = \begin{cases} \log p(\hat{\theta}_t)[A_t] + s_t \log \frac{3}{4} + (1-s_t) \log \frac{1}{4} + \log \frac{6}{10} & \text{if } A_t = a_t \\ \log p(\hat{\theta}_t)[A_t] + s_t \log \frac{1}{4} + (1-s_t) \log \frac{3}{4} + \log \frac{1}{10} & \text{otherwise} \end{cases}. \tag{9}$$