Causal Action Influence Aware Counterfactual Data Augmentation

Núria Armengol Urpí [1 2] **Marco Bagatella** [1 2] **Marin Vlastelica** [2] **Georg Martius** [3 2]

Abstract

Offline data are both valuable and practical resources for teaching robots complex behaviors. Ideally, learning agents should not be constrained by the scarcity of available demonstrations, but rather generalize beyond the training distribution. However, the complexity of real-world scenarios typically requires huge amounts of data to prevent neural network policies from picking up on spurious correlations and learning non-causal relationships. We propose CAIAC, a data augmentation method that can create feasible synthetic transitions from a fixed dataset without having access to online environment interactions. By utilizing principled methods for quantifying causal influence, we are able to perform counterfactual reasoning by swapping *action*-unaffected parts of the state-space between independent trajectories in the dataset. We empirically show that this leads to a substantial increase in robustness of offline learning algorithms against distributional shift. Videos, code and data are available at https://sites.google.com/view/caiac.

1. Introduction

Offline learning offers the opportunity of leveraging plentiful amounts of prerecorded data in situations where environment interaction is costly (Bahl et al., 2022; Brohan et al., 2022; 2023; Vlastelica et al., 2023). However, one of the fundamental challenges of such a framework is that of causal confusion.

Causal confusion arises when a trained agent misinterprets the underlying causal mechanics of the environment and, hence, fails to distinguish spurious correlations from causal links (De Haan et al., 2019; Gupta et al., 2023). When trying to reduce training loss, an agent can benefit from such spurious correlations in the data and, therefore, they can be inadvertently transferred to the mechanisms of learned models (Gupta et al., 2023).

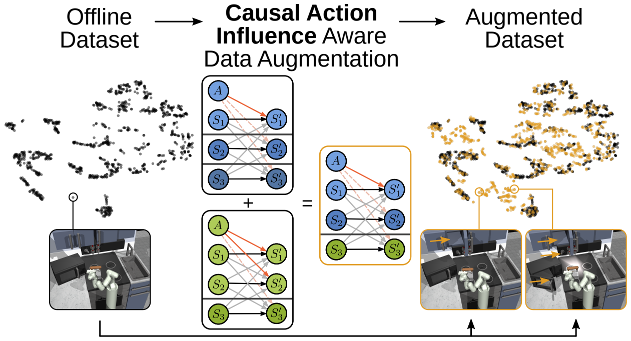


Figure 1: Overview of the proposed approach. Interactions between the agent and entities in the world are sparse. We use causal action influence (CAI), a local causal measure, to determine action-independent entities and create counterfactual data by swapping states of these entities from other observations in the dataset. Offline learning with these augmentations leads to better generalization.

Problematically, causally confused agents, are prone to catastrophic failure even in mild cases of distributional shift (De Haan et al., 2019), i.e. when the test distribution deviates from the training distribution. Subtle forms of distributional shift are common when learning from real-world data: collected demonstrations can only encompass a small fraction of the vast amount of possible configurations stemming from the inherent presence of many entities in the real world (Battaglia et al., 2018). Hence, at test times, agents are often queried on unfamiliar (i.e. out of distribution) configurations.

To illustrate the problem, let us imagine that we have demonstrations of a robot performing several kitchen-related activities: opening a microwave, sliding cabinets and turning on and off a light switch. However, the demonstrations only show how to perform those activities in this pre-specified order. Hence, following the example, the sliding cabinet (Event Y) is only shown to be open when the microwave (Event X) is open as well.

[1]Department of Computer Science, ETH Zurich, Zurich, Switzerland [2]Max Planck Institute for Intelligent Systems, Tübingen, Germany [3]Department of Computer Science, University of Tübingen, Tübingen, Germany. Correspondence to: Núria Armengol Urpí <nuriaa@ethz.ch>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

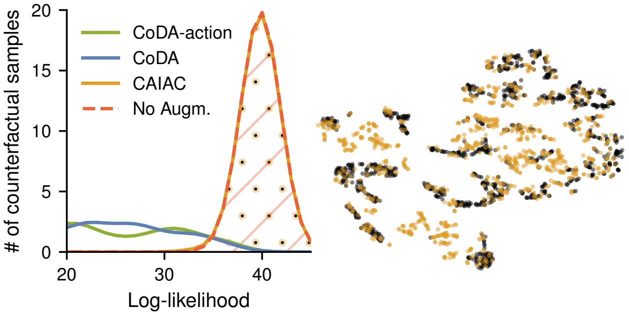


Figure 2: CAIAC counterfactual samples are consistent with the environment's dynamics and increase the support of the joint state space distribution, enabling the agent to be robust to distributional shift. Left: Log-likelihoods under the environment transition kernel of counterfactuals created with different methods. Right: Original data and counterfactuals augmentations with CAIAC visualized with t-SNE. Details on this evaluation are reported in Appendix A.7.

In this scenario, the trained agent happens to *simultaneously* observe independent events X and Y whenever it executes the sliding action (A), and hence may attribute the action A to X and Y occurring jointly, even though event X is independent of A. If the spurious correlation between X and Y observed during training fails to persist at test time, such a causally confused model may exhibit subpar performance. Namely, the agent might not be able to slide open the cabinet when the microwave is closed.

In contrast, humans are remarkably good at inferring what parts of the environment are relevant to solve a task, possibly due to relying on a causal representation of the world (Pearl & Mackenzie, 2018). This hypothesis has motivated the creation of *causal* approaches in machine learning that aim to identify relationships in the environment that will remain invariant under changes in the data-generating distribution. Existing work at the intersection between RL and causality has focused on an online learning (Lyle et al., 2021; Wang et al., 2022; Ding et al., 2023), imitation learning (De Haan et al., 2019) or partial observability setting (Forney et al., 2017; Kallus & Zhou, 2018). When learning in the online setting, some works operate in the interventional setting (Lyle et al., 2021; De Haan et al., 2019), i.e. a user may be able to "experiment" in the environment in order to discover causal structures by assigning values to some subset of the variables of interest and observing the effects on the rest of the system. In contrast, we focus on the challenging *offline* setting, where the agent is *not* capable of observing the real effects of such an intervention, and we propose an observational approach. While one could also learn to predict the outcome of the interventions by using a model-based approach, we note that an uninformed dynamics model can still be sensitive to spurious correlations and suffer from approximating errors.

Our approach, which we refer to as ***C**ausal **I**nfluence **A**ware **C**ounterfactual Data Augmentation* (CAIAC), introduces counterfactual data augmentations without the need for additional environment interactions, or reliance on counterfactual model rollouts. Instead, we exploit collected data to learn a causal model that explicitly reasons about causal influence and swap locally causally independent factors across different observed trajectories. Estimating the entire causal structure remains, however, a challenging task, particularly if attempted from offline data.

Taking this into account, we focus on identifying the effects the agent has on the environment; we thus assume action-influence to be more important for policy learning than potential object-object interactions. By partially trading off generality, this inductive bias on the underlying causal structure reduces the problem of estimating the full causal structure to only measuring the influence of actions over objects. Although causal discovery from observational data is known to be impossible for the general case (Pearl, 2009; Peters et al., 2017), methods exist that make use of some form of independence testing that have been successful in applications.

While other causal methods rely on heuristics to create new samples (Ding et al., 2023), on implicit measures for detecting causal influence among entities (Pitis et al., 2020) or on regularization of the dynamical models to suppress spurious correlations (Ding et al., 2023; Wang et al., 2021), our method is theoretically sound and relies on an *explicit* measure of influence, namely state-conditioned mutual information (Cover, 1999). We find this approach to be significantly more reliable at creating counterfactual samples that, not only follow the environment's dynamics (i.e. they are feasible), but also increase the support of the joint distribution over environment entities, as shown in Fig. 2.

Moreover, we show that by providing an offline learning agent with CAIAC's counterfactual samples, we prevent the agent from suffering from causal confusion, and we hence improve robustness to distributional shifts at test time. Our framework works as an independent module and can be used with any data-driven control algorithm. We demonstrate this through empirical results in high-dimensional offline goal-conditioned tasks, applying our method to fundamentally different data distributions and learning methods. Namely, we couple our method with offline goal-conditioned skill learning on the Franka-Kitchen environment (Gupta et al., 2019), and classical offline goal-conditioned reinforcement learning on Fetch-Push and FetchPick&Lift (Plappert et al., 2018). Across experiments, we show that CAIAC leads to enhanced performance in out-of-distribution settings and when learning from a modest amount of demonstrations.

2. Background

2.1. Reinforcement Learning

Markov Decision Processes (MDPs) are used as the basic formalism for sequential decision-making problems.

Definition 2.1 (Markov Decision Process (MDP)). A Markov Decision Process is a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \rho_0, \gamma)$, consisting of state space $\mathcal{S}$, action space $\mathcal{A}$, transition kernel $\mathcal{P}(S' \mid S, A)$, reward function $\mathcal{R} : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, initial state distribution ρ_0 and discount factor γ, respectively.

The goal of a learning algorithm is to extract a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ for maximizing the expected return $\mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)]$. Online algorithms have access to samples from the transition kernel $\mathcal{P}$, while offline methods leverage a fixed dataset $\mathcal{D}$ of trajectories which may be suboptimal. In this work, we focus on the offline setting, which particularly suffers from issues of distributional shift and out-of-distribution generalization.

2.2. Causal Graphical Models

Generally, a joint distribution $P(X)$ has a particular independence structure which induces a specific factorization. This independence structure is a consequence of the functional relationship (also called mechanism) between the variables that can be accurately described through a Structural Causal Model (SCM).

Definition 2.2 (SCM (Pearl, 2009)). A SCM is a tuple $(\mathcal{U}, \mathcal{V}, F, P^u)$, where $\mathcal{U}$ is a set of exogenous (unobserved) variables (e.g. the unobserved source of stochasticity in the environment) sampled from P^U, $\mathcal{V}$ is a set of endogenous (observed) variables (e.g. the observed state, the action and the reward in RL). F is the set of structural functions capturing the causal relations, such that functions $f_V : \mathrm{Pa}(V) \times \mathcal{U} \rightarrow V$ with $\mathrm{Pa}(V) \subset \mathcal{V}$ denoting the set of parents of V, determine the value of endogenous variables V for each $V \in \mathcal{V}$.

SCMs are usually visualized as a directed acyclic graph $\mathcal{G}$ whose nodes are associated with the variables in the SCM and whose edges indicate causal relationships. We say that a pair of variables v_i and v_j are confounded by a variable C (confounder) if they are both caused by C, i.e., $C \in \mathrm{Pa}(v_i)$ and $C \in \mathrm{Pa}(v_j)$. When two variables v_i and v_j do not have a direct causal link, they are still correlated if they are confounded, in which case this correlation is a *spurious correlation*. Given an SCM, one can make inferences about causal effects through the concept of an intervention.

Definition 2.3 (*do*-intervention (Pearl, 2009)). An intervention $do(V = v)$ on V induces a new SCM = $(\mathcal{U}, \mathcal{V}, F', P^u)$, where $F' = \{f_W \in F \mid W \neq V\} \cup \{f_{V=v}\}$ and $f_{V=v}(p, u) = v \; \forall p \in \mathrm{Pa}(V), u \in \mathcal{U}$.

An intervention on a set of nodes of the SCM effectively changes their structural equations, mostly replacing them by an explicit value. Interventional queries of the form $P(Y| \, do(X = x))$ are the so-called second rung of causation (Pearl, 2009). In this work we are interested in the third, counterfactual queries.

Definition 2.4 (Counterfactual). A counterfactual query is a query of the form $P(Y \mid do(X = x), \mathcal{U} = u)$, where $Y, X \subset \mathcal{V}$ and $\mathcal{U}$ is the set of exogenous variables of the underlying SCM.

A counterfactual query about variables Y is asking what would have happened to Y if under the same conditions $\mathcal{U} = u$ the intervention $do(X = x)$ had been performed.

3. Problem Definition

We assume a known and fixed state-space factorization $\mathcal{S} = \mathcal{S}_1 \times ... \times \mathcal{S}_N$ for N entities, where each factor $\mathcal{S}_i$ corresponds to the state of an entity. In practice, there are methods that allow to automatically determine the number of factors (Zaheer et al., 2017) and to learn latent representations of each entity (Burgess et al., 2019; Zadaianchuk et al., 2023; Locatello et al., 2020; Greff et al., 2019; Jiang et al., 2019; Seitzer et al., 2022). While we do not consider them for simplicity, our method can be applied on top of such techniques.

An MDP coupled with a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ induces an SCM describing the resulting trajectory distribution. Given the Markovian property of the MDP and flow of time, there only exist direct causal links $\{S_t, A_t\} \rightarrow S_{t+1}$ by definition, i.e. $S_{t+1} \perp\!\!\!\perp V \mid \{S_t, A_t\}$ for non-descendant nodes $V \notin \{S_t, A_t\}$. For our purposes, it suffices to look at the time slice sub-graph which is governed by the MDP transition kernel $\mathcal{P}$ between state S, action A and next state S' with state factors $\{S_i\}_{i=1}^N$.

In most non-trivial environments, there exists an edge $S_i/A \rightarrow S'_j$ for most i, j (Fig. 3(a)). However, interactions often become sparse once we observe a particular state configuration. We capture these local interactions by the notion of a *local causal model*.

Definition 3.1 (Local Causal Model (Pitis et al., 2020)). Given an SCM $(\mathcal{U}, \mathcal{V}, F, P^u)$, the local SCM induced by observing $V = v$ with $V \subset \mathcal{V}$ is the SCM with $F_{V=v}$ and the graph $\mathcal{G}_{do(V=v)}$ resulting from removing edges from $\mathcal{G}_{do(V=v)}$ until the graph is causally minimal.

We shall use shorthand $\mathcal{G}_v$ for $\mathcal{G}_{do(V=v)}$ and similar to reduce notational clutter. Where normally the vertex set $\{A\} \cup \{S'_j\}_{j=1}^N$ would be densely connected in the direction $A \rightarrow S'$, intervening on S results in a sparser causal dependency in $\mathcal{G}_s$. An example of this local causal structure is given in Fig. 3(b): the robot can only influence the kettle and its own end-effector through its actions, but none of the other entities. We will make heavy use of sparsity and locality in constructing a counterfactual data augmentation.

4. Method

The main challenge that we aim to tackle in this work is that of learning policies in the offline regime that are more robust to distributional shift, i.e. are not susceptible to spurious correlations. We achieve this by augmenting real data with counterfactual modifications to *causally action-unaffected* entities, hence creating samples outside the support of the data distribution. Our method relies on the observation that performing the intervention $do(S = s)$ reduces the number of edges between A and S' as per Definition 3.1, leaving some factors independent of A. Now, assuming that we observe the transition $S = s, A = a, S' = s'$ we pose the question of how we can do counterfactual reasoning without access to the true set of structural equations $F_{S=s}$, i.e. synthesize counterfactual transitions.

To this end, we need to learn the causal structure of the LCM, which is known to be a hard problem (Peters et al., 2017). Therefore, we make the key assumption that interactions between entities are sparse (i.e. only occur rarely) and are thus negligible. While the correctness of generated counterfactuals will rely on this assumption to hold, we argue that this is realistic in various robotics tasks. For example, in the kitchen environment depicted in Fig. 1, the entities can hardly influence each other. In fact, each entity is mostly controlled by the agent actions. This would also be the case in several manufacturing processes, in which interaction between entities should only happen under direct control of robots. Moreover, settings in which the assumption does not hold remain a significant challenge for most heuristic methods for causal discovery (Pitis et al., 2020), which underperform despite their generality. More formally, and in a graphical sense, we assume that there is no arrow $S_i \to S'_j, i \neq j$ as visualized by the gray dashed lines in Fig. 3(b). We note that only two groups of arrows remain in the causal graph: $S_j \to S'_j$, which we assume to always be present, and $A \to S'_j$. This practical assumption allows us to reduce the hard problem of local causal discovery to the more approachable problem of local action influence detection. As a result, instead of resorting to heuristics (Pitis et al., 2020), we make use of a more principled *explicit* measure of local influence, the Causal Action Influence (CAI) (Seitzer et al., 2021) measure, which we introduce below.

4.1. Causal Action Influence Detection

To predict the existence of the edge $A \to S'_j$ in the local causal graph $\mathcal{G}_s$, Seitzer et al. (2021) use conditional mutual information (CMI) (Cover, 1999) as a measure of dependence, which is zero if $S'_j \perp\!\!\!\perp A | S = s$. Therefore, in each state $S = s$ we use the point-wise CMI as a state-dependent quantity that measures causal action influence (CAI), given by

$$\begin{aligned} C^j(s) &:= I(S'_j; A \mid S = s) \\ &= \mathbb{E}_{a\sim\pi}\big[D_{KL}\big(P_{S'_j|S=s,A=a} \;\big|\big|\; P_{S'_j|S=s}\big)\big]. \end{aligned} \tag{1}$$

The transition model $P_{S'_j|S=s,A=a}$ is modeled as a Gaussian neural network (predicting mean and variance) which maximizes a log-likelihood objective. The conditional distribution $P_{S^j_{t+1}|S=s}$ is computed in practice using M empirical action samples with the full model: $P_{S'_j|S=s} \approx \frac{1}{M}\sum_{m=1}^{M} P_{S'_j|S=s,A=a^{(m)}},\ a^{(m)} \sim \pi$. The KL divergence in (1) can be estimated using an approximation for Gaussian mixtures from (Durrieu et al., 2012). We note that the transition model does not need to be queried autoregressively, which avoids the issue of compounding errors. We refer the reader to Seitzer et al. (2021) for more details.

4.2. Inferring Local Factorization

Having introduced the concepts of locality and object independence, as well as a method to detect causal action influence, we proceed to infer the local factorization which will be leveraged to create counterfactual experience. For each state s in our data set $\mathcal{D}$, we compute the uncontrollable set, as the set of entities in s for which the agent has no causal action influence, expressed as:

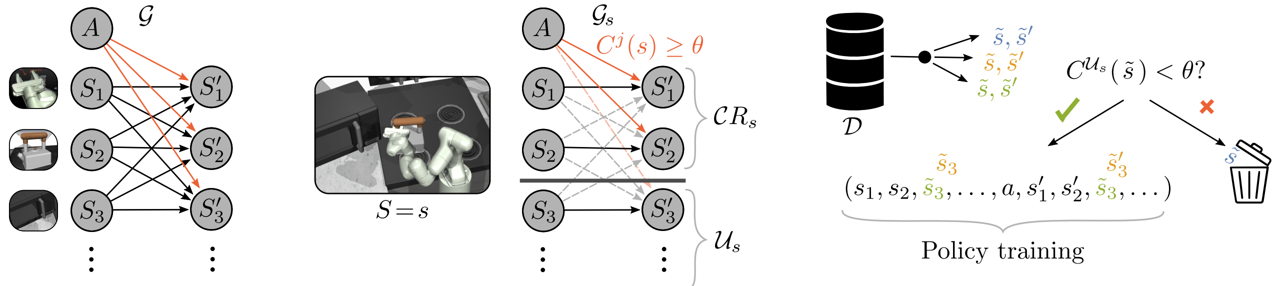
$$\mathcal{U}_s = \{s_j \mid C^j(s) \leq \theta, j \in [1, N]\} \tag{2}$$

where θ is a fixed threshold. The set $\mathcal{U}_s$ contains all entities j for which the arrow $A \to S'_j$ in the local causal graph $\mathcal{G}_s$ does not exist. The remaining entities are contained in the set of controllable entities $\mathcal{CR}_s = \{s_1, \ldots, s_N\} \setminus \mathcal{U}_s$. An illustration is given in Fig. 3(b).

With our assumptions and the sets $\mathcal{U}_s$ and $\mathcal{CR}_s$ we find that the local causal graph $\mathcal{G}_s$ is divided into the *disconnected* subgraphs $\mathcal{G}^{\mathcal{CR}}_s$, that contains the entities in $\mathcal{CR}$ and A, and into $|\mathcal{U}_s|$ *disconnected* subgraphs $\mathcal{G}^{\mathcal{U}}_{s_i}, i \in [1, |\mathcal{U}_s|]$, each of which contains an entity in $\mathcal{U}_s$ with only self-links, see Fig. 3(b). We can also compute the uncontrollable set for an extended time period κ, i.e. (I) $\mathcal{U}_{s_{t:t+\kappa}} = \bigcap_{\tau=t}^{t+\kappa-1} \mathcal{U}_{s_\tau}$.

4.3. Computing Counterfactuals

Given the partitioning of the graph described above, similarly to (Pitis et al., 2020), we can think of each subgraph as an independent causal mechanism that can be reasoned about separately. Assuming no unobserved exogenous variables, we may obtain counterfactuals in the following way: given two transitions (s, a, s') and $(\hat{s}, \hat{a}, \hat{s}') \in \mathcal{D}$ sampled for training, which have at least one uncontrollable subgraph structure in common (i.e. $\mathcal{U}_s \cap \mathcal{U}_{\hat{s}} \neq \emptyset$), we generate a counterfactual transition $(\tilde{s}, \tilde{a}, \tilde{s}')$ by swapping the entity transitions (s_i, s'_i) with $(\hat{s}_i, \hat{s}'_i)$ and $i \in \mathcal{U}_s \cap \mathcal{U}_{\hat{s}}$. In



a) Global causal graph

b) Local causal graph with object independence assumption

c) Counterfactual data augmentation from dataset $\mathcal{D}$

Figure 3: Illustration of counterfactual data augmentation. The global causal graph does not allow for factorization (a). Our local causal graph (b) is pruned by causal action influence. Object-object interactions are assumed to be rare/not existing (gray dashed). We swap elements that are not under control (i.e. in set $\mathcal{U}$) by samples from the data, thus creating counterfactual samples. We omit the exogenous variables from the global graph for compactness.

Algorithm 1: CAIAC

```
input Dataset $\mathcal{D}$
  Compute uncontrollable set $\mathcal{U}_s$, $\forall s \in \mathcal{D}$ (Eq. 2).
  Sample $(s, a, s') \sim \mathcal{D}$ and set $(\tilde{s}, \tilde{s}') \leftarrow (s, s')$
  for $s_i \in \mathcal{U}_s$
    Sample $(\hat{s}, \hat{a}, \hat{s}') \sim \mathcal{D}$
    if $\hat{s}_i \in \mathcal{U}_{\hat{s}}$ then $(\tilde{s}_i, \tilde{s}'_i) \leftarrow (\hat{s}_i, \hat{s}'_i)$
    yield $(s, a, s')$ and $(\tilde{s}, a, \tilde{s}')$
```

this way, we observe the result of the counterfactual query $P(S' \mid do(S = \tilde{s}, A = a))$ without using the mechanism $f_{S'}$ if it remains unchanged in the new LCM. However, even when only swapping the entity transitions for $\mathcal{U}_s \cap \mathcal{U}_{\hat{s}}$, the LCM resulting from the intervention $do(S = \tilde{s})$ may still contain a different mechanism $f_{S'}$ than the source LCM of s, meaning that the transition $(\tilde{s}, \tilde{a}, \tilde{s}')$ becomes invalid. An additional check of causal influence would entice an out-of-distribution query to the CAI measure, which is learned from transitions and therefore error-prone. In practice we do not perform this check and accept creating a small fraction of potentially infeasible transitions. The pseudocode of our method, which we call **Ca**usal **I**nfluence **A**ware **C**ounterfactual Data Augmentation (CAIAC), is given in Algorithm 1.

5. Related work

Data Augmentation Data augmentation is a fundamental technique for achieving improved sample-efficiency and generalization to new environments, especially in high-dimensional settings. In deep learning systems designed for computer vision, data augmentation can be found as early as in LeCun et al. (1998) and Krizhevsky et al. (2012), who leverage simple geometric transformations, such as random flips and crops. Naturally, a plethora of augmentation techniques (Berthelot et al., 2019; Sohn et al., 2020) have been proposed over time. To improve generalization in RL, domain randomization (Tobin et al., 2017; Pinto et al., 2017) is often used to transfer policies from simulation to the real world by utilizing diverse simulated experiences. Cobbe et al. (2019); Lee et al. (2019) showed that simple augmentation techniques, such as cutout and random convolution, can be useful to improve generalization in RL from images. Similarly to us, (Laskin et al., 2020) use data augmentation for RL without any auxiliary loss. Crucially, most data augmentations techniques in the literature require human knowledge to augment the data according to domain-specific invariances (e.g., through cropping, rotation, or color jittering), and mostly target the learning from image settings. Nevertheless, heuristics for data augmentation can be formally justified through a causal invariance assumption with respect to certain perturbation on the inputs.

Offline learning, distributional shift and causal confusion Offline RL and imitation learning methods rely on the availability of informative demonstrations (Lange et al., 2012; Li et al., 2023; Urpí et al., 2023; Lynch et al., 2019; Vlastelica et al., 2021b). However, in low-data regimes or task-agnostic demonstrations settings, these methods often suffer from distributional shift. This shift occurs when the agent induces a state-action distribution which deviates from the original data (Ross et al., 2011). Several offline learning methods have been proposed for fighting distributional shift, such as by minimizing deviation from the behavior policy (Fujimoto et al., 2019; Kumar et al., 2019; Kostrikov et al., 2021; Urpí et al., 2021) or minimizing risk (Vlastelica et al., 2021a). In imitation learning, several works focus on solving the causal confusion problem (De Haan et al., 2019), where a policy exploits nuisance correlates in the states for predicting expert actions (Wen et al., 2020; Seo et al., 2024).

Causal Reinforcement Learning Detecting causal influence involves causal discovery, which can be pictured as finding the existence of arrows in a causal graph (Pearl, 2009). While it remains an unsolved task in its broadest sense, there are assumptions that permit discovery in some settings (Peters et al., 2012; Spirtes et al., 2001).

Once the existence of an arrow can be detected, its impact needs to be established, for which several measures, such as transfer entropy or information flow, have been proposed (Schreiber, 2000; Lizier, 2012; Ay & Polani, 2008). In our case, we use conditional mutual information (Cover, 1999) as a measure of causal action influence, as proposed by Seitzer et al. (2021).

The intersection of RL and causality has recently been studied to improve interpretability, sample efficiency, and to learn better representations (Buesing et al., 2018; Bareinboim et al., 2015; Lu et al., 2018; Rezende et al., 2020).

Lyle et al. (2021) dealt with the problem of causal hypothesis-testing in the online setting via an exploration algorithm, Ding et al. (2023) utilize a model-based counterfactual approach (Pitis et al., 2022) for solving a robust MDP and Zhang et al. (2020) investigate how to obtain a reduced causal graph using a block structure. While Wang et al. (2022) also leverage influence to learn the causal structure, they use it for state abstraction to improve model generalization. Similarly to us, (Lu et al., 2020; Ding et al., 2022), also provide the agent with counterfactuals, but by learning and querying a model for the state transition. In particular, our work is related to that of Pitis et al. (2020), which proposes the Local Causal Model framework to generate counterfactual data, and underpins our work. However, Pitis et al. (2020) aim at estimating the entire local causal graph, which is a challenging problem. In practice, they rely on a heuristic method based on the attention weights of a transformer world model, which does not scale well to high-dimensional environments. In contrast, our method does not require learning the entire local causal graph, as it assumes that the interactions between entities (except the agent) are sparse enough to be neglected. This also implies that the agent is the only entity that can influence the rest of the entities through its actions. Therefore, this setting is related to the concept of contingency awareness from psychology (Watson, 1966), which was interestingly already considered in deep reinforcement learning methods for Atari (Song et al., 2020; Choi et al., 2018).

6. Experiments

We evaluate CAIAC in two goal-conditioned settings: offline RL and offline self-supervised skill learning. In particular, we are interested in evaluating whether CAIAC **(i)** leads to better robustness to extreme distributional shifts, **(ii)** enlarges the support of the joint distribution over the state space in low data regimes, and **(iii)** works as an independent module combinable with arbitrary learning-based control algorithms.

The set of benchmarks revolves around the issue of spurious correlations in the state space in manipulation domains, and is built upon the Franka-Kitchen (Gupta et al., 2019) and the Fetch (Plappert et al., 2018) platforms. The training data for each task includes spurious correlations that can usually appear during the data collection process, and could distract the policy from learning important features of the state.

Baselines We compare CAIAC with CoDA (Pitis et al., 2020), a counterfactual data augmentation method, which uses the attention weights of a transformer model to estimate the local causal structure. Given two transitions that share local causal structures, it swaps the connected components to form new transitions. Additionally, we compare with an ablated version of CoDA, CoDA-ACTION, which only estimates the influences of the action using the transformer weights and thus is a 'heuristic'-sibling of our method. For the RL experiments, we also compare it with RSC (Ding et al., 2023), a framework for robust reinforcement learning that constructs new samples by perturbing the value of the states using an heuristic and learns a structural causal model to predict the next state given the perturbed state. As an ablation, we include a baseline without data augmentation (NO-AUGM). For completeness, we provide comparisons with model-based approaches in B.2.

Given our method and some of the baselines performances depend on an appropriate choice of the parameter θ to get a classification of influence, we provide a thorough analysis on how this parameter was chosen in Appendix A.5.

6.1. Tasks with Spurious Correlations

Our initial experiments investigate whether CAIAC can increase the generalization capabilities of algorithms when learning from demonstrations that include spurious correlation. Specifically we test whether the trained algorithms are robust to extreme state distributional shifts at test time.

6.1.1. GOAL CONDITIONED OFFLINE SELF-SUPERVISED SKILL LEARNING

We apply our method to the challenging Franka-Kitchen environment from Gupta et al. (2019). We make use of the data provided in the D4RL benchmark (Fu et al., 2020), which consists of a series of teleoperated sequences in which a 7-DoF robot arm manipulates different parts of the environment (e.g., it opens microwave, switches on the stove). Crucially, all demonstrations are limited to a few manipulation sequences (for example, first opening the microwave, turning on a burner, and finally the light). Thus, the support of the joint distribution over entities in the environment is reduced to only a few combinations. To illustrate this with an example, the light is only on when the microwave is open. At test time we evaluate the trained agent in unseen

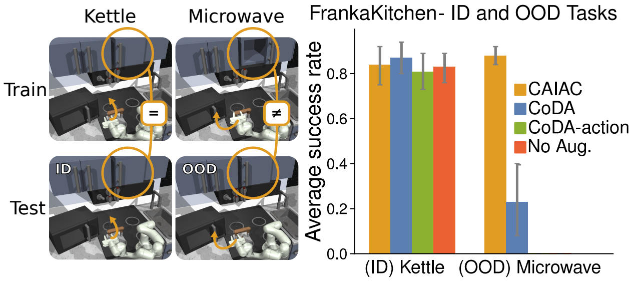


Figure 4: Motivating Franka-Kitchen example. The experimental setup (left) and success rates for in-distribution and out-of-distribution tasks (right). Metrics are averaged over 10 seeds and 10 episodes per task, with 95% simple bootstrap confidence intervals.

configurations, breaking the spurious correlations that exist in the training data. We hypothesize that CAIAC will create valid counterfactual data such that the downstream learning algorithms would be able to generalize to unseen state configurations. As a downstream learning algorithm we use LMP (Lynch et al., 2019), an offline goal-conditioned self-supervised learning algorithm, which learns to map similar behaviors (or state-action trajectories) into a latent space from which goal-conditioned plans can be sampled. Formally, LMP is a sequence-to-sequence VAE (Sohn et al., 2015; Bowman et al., 2015) autoencoding random experiences extracted from the dataset through a latent space. In our case, we use experiences of fixed window length κ. Given the inherent temporal abstraction of the algorithm, we generate counterfactuals of fixed length $\kappa > 1$ by computing the uncontrollable set for the entire window as the intersection over all time slices, as in ((I)). For specific details on the learning algorithm and the Franka-Kitchen environment, we refer to A.1.1 and A.2.1 respectively.

Franka-Kitchen: A Motivating Experiment Our first experiment is designed to verify claim (i), i.e., that CAIAC enables generalization to unseen configurations over entities. First, we showcase this in a simple and controlled environment. Thus, we create a reduced modified dataset from the original D4RL dataset (Fu et al., 2020), that contains only demonstrations for the microwave task (MW) and the kettle (K) task. During demonstrations for the (MW) task, we initialize the cabinet to be always open, whereas for demonstrations for the (K) task, it remains closed. The rest of the objects are set to the default initial configuration (see A.2.1). At inference time, we initialize the environment with its default initial configuration (crucially, the cabinet is closed), and we evaluate both tasks ((K) and (MW)), as shown in Fig. 4(left). Hence, while the (K) task was demonstrated for the current configuration (in-distribution, ID), the agent is effectively evaluated on an out-of-distribtion (OOD) configuration for the (MW) task.

We evaluate success rate on both tasks with CAIAC and all baselines, as shown in Fig. 4(right). All methods are able to solve the (K) task, as expected, since it is in-distribution (ID). However, we observe fundamentally different results for the OOD (MW) task. In principle, CAIAC can detect that the sliding cabinet is never under control of the agent, and will be able to create the relevant counterfactuals to prevent the policy from picking up on the spurious correlation in the data. Indeed, the performance of CAIAC in the OOD task (MW) is not affected, and it is the same as for the ID task. On the other hand, the performance of CoDA and CoDA-action is drastically impaired in the OOD setting. Despite the simplicity of the setting, the input dimensionality of the problem is high, and the transformer attention weights are not able to recover the correct causal graph. By picking up on spurious correlations, the attention weights of the transformer estimate low influence from the action to all entities (even the agent), and hence CoDA-action creates dynamically-unfeasible counterfactuals which affect performance. Since the ratio of observed-counterfactuals data is 1:1 we hypothesize that there is enough in-distribution data to not affect the (K) task for CoDA-action. The local graph induced by CoDA has at least as many edges as the one of CoDA-action, and hence the probability for creating unfeasible counterfactuals is lower. We hypothesize, that despite not learning correct causal influence, it might still provide some samples which benefit the learning algorithm and allow for an average OOD success rate of 0.2. We refer the reader to Appendix A.3 for further analysis on the impact of the ratio of observed:counterfactual data for this experiment. Finally, as expected, No Augm. fails to solve the OOD (MW) task.

Franka-Kitchen: All Tasks Having evaluated CAIAC in a controlled setting, we now scale up the problem to the entire Franka-Kitchen D4RL dataset. While in the standard benchmark the agent is required to execute a single fixed sequence of tasks, we train a goal-conditioned agent and evaluate on the full range of tasks, which include the microwave, the kettle, the slider, the hinge cabinet, the light switch and the bottom left burner tasks (Mendonca et al., 2021). One task is sampled for each evaluation episode. While alleviating the need for long-horizon planning, this results in a challenging setting, as only a subset of tasks is shown directly from the initial configuration. However, the largest challenge in our evaluation protocol lies in the creation of unobserved state configurations at inference time. While the provided demonstrations always start from the same configuration (e.g., the microwave is always initialized as closed), at inference time, we initialize all non-target entities (with $p = 0.5$) to a random state, hence exposing the agent to OOD states. We expect that agents trained with

Table 1: Average success rates for Franka-Kitchen tasks with OOD initial configurations, computed over 10 seeds and 20 episodes per task with 90% simple bootstrap confidence intervals.

Algorithm	CAIAC	CoDA	CoDA-action	No-Augm.
Kettle	**0.81** ± **0.07**	0.18 ± 0.05	0.16 ± 0.07	0.07 ± 0.06
Microwave	**0.75** ± **0.09**	0.07 ± 0.05	0.0 ± 0.03	0.01 ± 0.03
Bottom-burner	**0.13** ± **0.05**	0.01 ± 0.03	0.0 ± 0.02	0.01 ± 0.02
Slide cabinet	**0.14** ± **0.04**	0.1 ± 0.03	0.02 ± 0.02	0.07 ± 0.03
Light switch	**0.01** ± **0.01**	0.0 ± 0.0	0.0 ± 0.0	0.00 ± 0.0
Hinge cabinet	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0

CAIAC will show improved performance to unseen environment configurations, as those can be synthesized through counterfactual data augmentation. The results, shown in Table 1, are consistent with the challenging nature of this benchmark, as the evaluated tasks involve OOD settings in terms of states and actions. Nevertheless, we find that CAIAC is significantly better than baselines in 6/7 tasks, while the last task is unsolved by methods. We hypothesize that the low performance on some of the tasks is due to the absence of robot state and action trajectories in the dataset that show how to solve each of the tasks from the initial robot joint configuration. Hence, even with perfect counterfactual data augmentation these tasks remain challenging. We refer the reader to the Appendix A.2.1 for further analysis. As observed in the simplified setting, methods relying on heuristic-based causal discovery (CoDA and CoDA-action) suffer from misestimation of causal influence, and thus from the creation of dynamically-unfeasible training samples. See A.7 for a further analysis on the quality of the created counterfactuals and Fig. 7 for a visualization of the computed CAI scores per each entity on one of the demonstrations for the Franka-Kitchen dataset. Finally, without any data augmentation, the learning algorithm i.e. No Augm. baseline) fails to perform the OOD tasks.

6.1.2. Goal-conditioned Offline RL

Fetch-Pick&Lift with 4 cubes We additionally test CAIAC on Fetch-Pick&Lift, a modified version of the Fetch-Pick&Place environment (Plappert et al., 2018) were a robot needs to pick and lift a desired cube out of 4 arranged on a table (Fig. 5 (left)). For this benchmark, we include spurious correlations in the training data by always arranging the cubes in a line. At test time, the cubes are randomly positioned on the table, evaluating the agent in out of distribution states. We collect 40k trajectories using an expert policy (50%) and random policy (50%) and train an agent offline using TD3+BC (Fujimoto & Gu, 2021).

Results are shown in Fig. 5 (left). We observe that CAIAC reaches a high success rate by creating relevant counterfactuals that augment the support of the joint distribution over entities and break the spurious correlations in the data. Conversely, the rest of the baselines exhibit subpar performance. Once again, for CoDA and CoDA-action, the attention weights of the transformer fail to recover the correct causal graph, resulting in the generation of infeasible samples. On the other hand, the causally uniformed heuristics used to perturb the states in RSC, might break the true cause and effect relationships between state dimensions, leading to performance drop, as reported in (Ding et al., 2023). Moreover, there is no theoretical guarantee that the learned dynamics model, which is regularized for improved generalization, is inherently causal. Consequently, the generated next-state samples may be inaccurate. Further details are given in Appendices A.1.3 and A.2.3.

6.2. Low Data Regimes

6.2.1. Goal-conditioned Offline RL

With this final experiment, our aim is to verify claim (ii), i.e., that CAIAC can enlarge the support of the joint distribution in low data regimes, even when spurious correlations are not necessarily present in the training data, and no distributional shift is injected at test time.

Fetch-Push with 2 cubes We evaluate CAIAC in a Fetch-Push environment (Plappert et al., 2018), where a robotic arm has to slide two blocks to target locations. For this experiment we collect 20k trajectories using an expert policy (30%) and random policy (70%) and train an agent offline using TD3 (Fujimoto et al., 2018) in two data regimes: namely 100% and 20% of data.

More details are given in Appendices A.1.2, A.2.2 and A.5. We compare success rates between baselines and CAIAC among different data regimes in Fig. 5 (right).

In the high data regime, CAIAC and No Augm. baseline perform similarly given that there is enough coverage of the state space in the original dataset. In contrast, in the low data regime CAIAC performs significantly better. Given that the samples in the data, cover sufficient support of the marginal distribution of each entity, CAIAC can substantially increase the support of the joint distribution over entities, leading to higher performance. Transformer-based methods CoDA and CoDA-action, and RSC create detrimental counterfactuals in all data regimes leading to decreased performance. To showcase the previous claims, the estimated influence scores for all the methods are visualized in A.4. We note that, while previous work (Pitis et al., 2020) has shown good online performance of CoDA in this environment, it resorted to a handcrafted heuristic to decide about influence.

7. Discussion

While extracting complex behaviors from pre-collected datasets is a promising direction for robotics, data scarcity

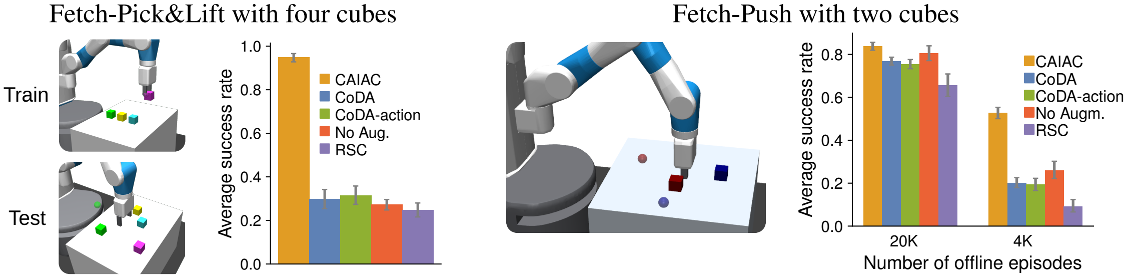


Figure 5: Success rates for Fetch-Pick&Lift with 4 objects (left) and Fetch-Push with 2 cubes (right). Metrics are averaged over 30 seeds and 50 episodes with 95% simple bootstrap confidence intervals.

remains a principal issue in high-dimensional, multi-object settings, due to a combinatorial explosion of possible state configurations which cannot be covered densely by demonstrations. Hence, current learning methods often pick up on spurious correlations and struggle to generalize to unseen configurations. In this paper, we propose CAIAC as a method for counterfactual data augmentation without the need for additional environment interaction or counterfactual model rollouts, which can be used with any learning algorithm. By adding an inductive bias on the causal structure of the graph, we circumvent the problem of full causal discovery and reduce it to the computation of an explicit measure of the agent's causal action influence over objects. Empirically, we show that CAIAC leads to enhanced performance and generalization to unseen configurations, suggesting that further advances in addressing both partial and full causal discovery problems can be substantially beneficial for robot learning. While our current approach deems action influence to be more important than object-object interaction, in future work, we aim to explore alternative forms of independence to make our approach applicable to a wider range of tasks. Finally, we would like to further investigate rebalancing the data distribution to counteract data imbalances in the dataset.

Reproducibility Statement In order to ensure reproducibility of our results, we make our codebase publicly available at https://sites.google.com/view/caiac, and provide detailed instructions for training and evaluating the proposed method. Furthermore, we describe algorithms and implementation details in Appendix A. Finally, as our experiments rely on offline datasets, we publish them at the same link.

Acknowledgements

The authors thank Cansu Sancaktar and Max Seitzer for their help reviewing the manuscript and the annonymous reviewers for their valuable feedback. The authors thank the Max Planck ETH Center for Learning Systems for supporting Núria Armengol and Marco Bagatella, and the International Max Planck Research School for Intelligent Systems for supporting Marin Vlastelica. Georg Martius is a member of the Machine Learning Cluster of Excellence, funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC number 2064/1 – Project number 390727645. This work was supported by the ERC - 101045454 REAL-RL.

Impact Statement

This paper aims to push the boundaries of generalization in machine learning. Hence, it shares the many societal consequences tied to the field of machine learning, from ethical to environmental consequences.

References

Andrychowicz, M., Wolski, F., Ray, A., Schneider, J., Fong, R., Welinder, P., McGrew, B., Tobin, J., Pieter Abbeel, O., and Zaremba, W. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Ay, N. and Polani, D. Information flows in causal networks. *Advances in complex systems*, 11(01):17–41, 2008.

Bahl, S., Gupta, A., and Pathak, D. Human-to-robot imitation in the wild. In *Robotics Science and Systems (RSS)*, 2022.

Bareinboim, E., Forney, A., and Pearl, J. Bandits with unobserved confounders: A causal approach. *Advances in Neural Information Processing Systems*, 28, 2015.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., et al. Relational inductive biases, deep learning, and graph networks. arXiv preprint arXiv:1806.01261, 2018.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A., and Raffel, C. A. Mixmatch: A holistic approach to semi-supervised learning. *Advances in neural information processing systems*, 32, 2019.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Chen, X., Choromanski, K., Ding, T., Driess, D., Dubey, A., Finn, C., et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

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

Burgess, C. P., Matthey, L., Watters, N., Kabra, R., Higgins, I., Botvinick, M., and Lerchner, A. Monet: Unsupervised scene decomposition and representation, 2019.

Choi, J., Guo, Y., Moczulski, M., Oh, J., Wu, N., Norouzi, M., and Lee, H. Contingency-aware exploration in reinforcement learning. *CoRR*, abs/1811.01483, 2018. URL http://arxiv.org/abs/1811.01483.

Cobbe, K., Klimov, O., Hesse, C., Kim, T., and Schulman, J. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pp. 1282–1289. PMLR, 2019.

Cover, T. M. *Elements of information theory*. John Wiley & Sons, 1999.

De Haan, P., Jayaraman, D., and Levine, S. Causal confusion in imitation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Ding, W., Lin, H., Li, B., and Zhao, D. Generalizing goal-conditioned reinforcement learning with variational causal reasoning. *Advances in Neural Information Processing Systems*, 35:26532–26548, 2022.

Ding, W., Shi, L., Chi, Y., and Zhao, D. Seeing is not believing: Robust reinforcement learning against spurious correlation. *arXiv preprint arXiv:2307.07907*, 2023.

Durrieu, J.-L., Thiran, J.-P., and Kelly, F. Lower and upper bounds for approximation of the kullback-leibler divergence between gaussian mixture models. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4833–4836, 2012. doi: 10.1109/ICASSP.2012.6289001.

Forney, A., Pearl, J., and Bareinboim, E. Counterfactual data-fusion for online reinforcement learners. In *International Conference on Machine Learning*, pp. 1156–1164. PMLR, 2017.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4rl: Datasets for deep data-driven reinforcement learning, 2020.

Fujimoto, S. and Gu, S. S. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Fujimoto, S., Meger, D., and Precup, D. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Greff, K., Kaufman, R. L., Kabra, R., Watters, N., Burgess, C., Zoran, D., Matthey, L., Botvinick, M., and Lerchner, A. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

Gupta, A., Kumar, V., Lynch, C., Levine, S., and Hausman, K. Relay policy learning: Solving long horizon tasks via imitation and reinforcement learning. *Conference on Robot Learning (CoRL)*, 2019.

Gupta, G., Rudner, T. G., McAllister, R. T., Gaidon, A., and Gal, Y. Can active sampling reduce causal confusion in offline reinforcement learning? In *Conference on Causal Learning and Reasoning*, pp. 386–407. PMLR, 2023.

Hafner, D., Lillicrap, T., Norouzi, M., and Ba, J. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020.

Jiang, J., Janghorbani, S., De Melo, G., and Ahn, S. Scalor: Generative world models with scalable object representations. *arXiv preprint arXiv:1910.02384*, 2019.

Kallus, N. and Zhou, A. Confounding-robust policy improvement. *Advances in neural information processing systems*, 31, 2018.

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

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Kumar, A., Fu, J., Soh, M., Tucker, G., and Levine, S. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Lange, S., Gabel, T., and Riedmiller, M. Batch reinforcement learning. In *Reinforcement learning: State-of-the-art*, pp. 45–73. Springer, 2012.

Laskin, M., Lee, K., Stooke, A., Pinto, L., Abbeel, P., and Srinivas, A. Reinforcement learning with augmented data. *Advances in neural information processing systems*, 33: 19884–19895, 2020.

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

Lee, K., Lee, K., Shin, J., and Lee, H. Network randomization: A simple technique for generalization in deep reinforcement learning. *arXiv preprint arXiv:1910.05396*, 2019.

Li, C., Vlastelica, M., Blaes, S., Frey, J., Grimminger, F., and Martius, G. Learning agile skills via adversarial imitation of rough partial demonstrations. In Liu, K., Kulic, D., and Ichnowski, J. (eds.), *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 342–352. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/li23b.html.

Lizier, J. T. *The local information dynamics of distributed computation in complex systems*. Springer Science & Business Media, 2012.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Lu, C., Schölkopf, B., and Hernández-Lobato, J. M. Deconfounding reinforcement learning in observational settings. *arXiv preprint arXiv:1812.10576*, 2018.

Lu, C., Huang, B., Wang, K., Hernández-Lobato, J. M., Zhang, K., and Schölkopf, B. Sample-efficient reinforcement learning via counterfactual-based data augmentation. *arXiv preprint arXiv:2012.09092*, 2020.

Lyle, C., Zhang, A., Jiang, M., Pineau, J., and Gal, Y. Resolving causal confusion in reinforcement learning via robust exploration. In *Self-Supervision for Reinforcement Learning Workshop-ICLR*, volume 2021, 2021.

Lynch, C., Khansari, M., Xiao, T., Kumar, V., Tompson, J., Levine, S., and Sermanet, P. Learning latent plans from play. *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1903.01973.

Mendonca, R., Rybkin, O., Daniilidis, K., Hafner, D., and Pathak, D. Discovering and achieving goals via world models. *Advances in Neural Information Processing Systems*, 34:24379–24391, 2021.

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

Pearl, J. and Mackenzie, D. *The book of why: the new science of cause and effect*. Basic books, 2018.

Peters, J., Mooij, J. M., Janzing, D., and Schölkopf, B. Identifiability of causal graphs using functional models. *CoRR*, abs/1202.3757, 2012. URL http://arxiv.org/abs/1202.3757.

Peters, J., Janzing, D., and Schölkopf, B. *Elements of Causal Inference: Foundations and Learning Algorithms*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2017. ISBN 978-0-262-03731-0. URL https://mitpress.mit.edu/books/elements-causal-inference.

Pinto, L., Andrychowicz, M., Welinder, P., Zaremba, W., and Abbeel, P. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

Pitis, S., Creager, E., and Garg, A. Counterfactual data augmentation using locally factored dynamics. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

Pitis, S., Creager, E., Mandlekar, A., and Garg, A. Mocoda: Model-based counterfactual data augmentation. *Advances in Neural Information Processing Systems*, 35:18143–18156, 2022.

Plappert, M., Andrychowicz, M., Ray, A., McGrew, B., Baker, B., Powell, G., Schneider, J., Tobin, J., Chociej, M., Welinder, P., et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

Rezende, D. J., Danihelka, I., Papamakarios, G., Ke, N. R., Jiang, R., Weber, T., Gregor, K., Merzic, H., Viola, F., Wang, J., et al. Causally correct partial models for reinforcement learning. *arXiv preprint arXiv:2002.02836*, 2020.

Rosete-Beas, E., Mees, O., Kalweit, G., Boedecker, J., and Burgard, W. Latent plans for task agnostic offline reinforcement learning. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

Ross, S., Gordon, G., and Bagnell, D. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 627–635. JMLR Workshop and Conference Proceedings, 2011.

Schreiber, T. Measuring information transfer. *Physical review letters*, 85(2):461, 2000.

Seitzer, M., Schölkopf, B., and Martius, G. Causal influence detection for improving efficiency in reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS 2021)*, December 2021. URL https://arxiv.org/abs/2106.03443.

Seitzer, M., Horn, M., Zadaianchuk, A., Zietlow, D., Xiao, T., Simon-Gabriel, C.-J., He, T., Zhang, Z., Schölkopf, B., Brox, T., et al. Bridging the gap to real-world object-centric learning. *arXiv preprint arXiv:2209.14860*, 2022.

Seo, S., Hwang, H., Yang, H., and Kim, K.-E. Regularized behavior cloning for blocking the leakage of past action information. *Advances in Neural Information Processing Systems*, 36, 2024.

Sohn, K., Lee, H., and Yan, X. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015.

Sohn, K., Berthelot, D., Carlini, N., Zhang, Z., Zhang, H., Raffel, C. A., Cubuk, E. D., Kurakin, A., and Li, C.-L. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.

Song, Y., Wang, J., Lukasiewicz, T., Xu, Z., Zhang, S., Wojcicki, A., and Xu, M. Mega-reward: Achieving human-level play without extrinsic rewards. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5826–5833, Apr. 2020. doi: 10.1609/aaai.v34i04.6040. URL https://ojs.aaai.org/index.php/AAAI/article/view/6040.

Spirtes, P., Glymour, C., and Scheines, R. *Causation, prediction, and search*. MIT press, 2001.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 23–30. IEEE, 2017.

Urpí, N. A., Curi, S., and Krause, A. Risk-averse offline reinforcement learning. *arXiv preprint arXiv:2102.05371*, 2021.

Urpí, N. A., Bagatella, M., Hilliges, O., Martius, G., and Coros, S. Efficient learning of high level plans from play. In *International Conference on Robotics and Automation (ICRA)*, 2023.

Vlastelica, M., Blaes, S., Pinneri, C., and Martius, G. Risk-averse zero-order trajectory optimization. In *5th Annual Conference on Robot Learning*, 2021a.

Vlastelica, M., Rolinek, M., and Martius, G. Neuro-algorithmic policies enable fast combinatorial generalization. In *International Conference on Machine Learning*, pp. 10575–10585. PMLR, 2021b.

Vlastelica, M., Cheng, J., Martius, G., and Kolev, P. Diverse offline imitation learning, 2023.

Wang, Z., Xiao, X., Zhu, Y., and Stone, P. Task-independent causal state abstraction. In *Proceedings of the 35th International Conference on Neural Information Processing Systems, Robot Learning workshop*, 2021.

Wang, Z., Xiao, X., Xu, Z., Zhu, Y., and Stone, P. Causal dynamics learning for task-independent state abstraction. *arXiv preprint arXiv:2206.13452*, 2022.

Watson, J. S. The development and generalization of "contingency awareness" in early infancy: Some hypotheses. *Merrill-Palmer Quarterly of Behavior and Development*, 12(2):123–135, 1966. ISSN 00260150. URL http://www.jstor.org/stable/23082793.

Wen, C., Lin, J., Darrell, T., Jayaraman, D., and Gao, Y. Fighting copycat agents in behavioral cloning from observation histories. *Advances in Neural Information Processing Systems*, 33:2564–2575, 2020.

Zadaianchuk, A., Seitzer, M., and Martius, G. Object-centric learning for real-world videos by predicting temporal feature similarities, 2023.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Zhang, A., McAllister, R. T., Calandra, R., Gal, Y., and Levine, S. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2020.

A. Appendix

A.1. Implementation of downstream learning algorithms

In this section, we report implementation details concerning the learning algorithms. For a fine-grained description of all hyperparameters, we refer to our codebase at `https://sites.google.com/view/caiac`.

A.1.1. Goal-conditioned offline self-supervised skill learning

For the goal-conditioned self-supervised learning experiments we used LMP (Lynch et al., 2019), a goal-conditioned self-supervised method. It consists of a stochastic sequence encoder, or learned posterior, which maps a sequence τ to a distribution in latent plan space $q(z|\tau)$, a stochastic encoder or learned goal-conditioned prior $p(z|s, g)$ and a decoder or plan and goal conditioned policy: $\pi(a|z, s, g)$. The self-supervised goals g are relabeled from achieved goals in the trajectory.For counterfactual samples, trajectories are augmented *before* goal sampling. The main difference with the original implementation is that the latent goal representation is only added to the prior, but not the decoder. Additionally, we also implemented KL balancing in the loss term between the learned prior and the posterior: we minimize the KL-loss faster with respect to the prior than the posterior. Given that the KL-loss is bidirectional, in the beginning of training, we want to avoid regularizing the plans generated by the posterior towards a poorly trained prior. Hence, we use different learning rates, $\alpha = 0.8$ for the prior and $1-\alpha$ for the posterior, similar to Hafner et al. (2020). These two modifications were also suggested in (Rosete-Beas et al., 2022). Additionally, our decoder was open-loop (instead of close loop): given a sampled latent plan z it decodes the whole trajectory of length `skill length`$= N$, i.e. our decoder is $\pi(\hat{a}|z)$, where $\hat{a} = a_t, ..., a_{t+N}$ is the sequence of decoded actions, instead of $\pi(a|z, s, g)$. This modification was needed due to the *skewness* of the dataset. Since the demonstrations were provided from an expert agent, given most of the states, the distribution over actions is unimodal: when the robot is close to the microwave, the only sequence of actions in the dataset is the one that opens the microwave. Hence, a close-loop decoder would learn to ignore the latent plan, and only rely on the state. To solve this issue, we make the decoder open-loop.

A.1.2. Goal-conditioned Offline RL: TD3

We implement the TD3 algorithm (Fujimoto et al., 2018) with HER (Andrychowicz et al., 2017). Unless specified differently, the hyperparameters used were the ones from the original TD3 implementation. We use HER (Andrychowicz et al., 2017) to relabel the goals for real data, with a `future` relabeling strategy with $p = 0.5$, where the time points were sampled from a geometric distribution with $p_{geom} = 0.2$. For the counterfactual data we relabel the goals with $p = 0.5$ random sampling from the achieved goals in the buffer of counterfactual samples. In the experiments, we realized that the relabeling strategy had an impact on the performance of the downstream agent. To disentangle the impact of the relabeling strategy from the impact of the counterfactual data generation and to ensure a fair comparison, we also relabeled the same percentage of goals (i.e. $p = 0.25$) with `random` strategy for the `No Augm.` baseline. We train each method for 1.2M gradient steps, although all methods reach convergence after 600k gradient steps. For all baselines, the percentage of counterfactuals in each batch is set to 0.5.

A.1.3. Goal-conditioned Offline RL: TD3+BC

We implement the TD3+BC algorithm (Fujimoto & Gu, 2021) with HER (Andrychowicz et al., 2017) where a weighted behavior cloning loss is added to the policy update. After tuning, we used $\alpha_{BC} = 2.5$ ($\alpha_{BC} \to 1$ recovers Behavior Cloning, while $\alpha_{BC} \to 0$ recovers RL). Unless specified differently, the rest of hyperparameters used were the ones from the original TD3 implementation. We use HER (Andrychowicz et al., 2017) to relabel the goals for real data, with a `future` relabeling strategy with $p = 0.5$, where the time points were sampled from a geometric distribution with $p_{geom} = 0.2$ and

For the counterfactual data we relabel the goals with $p = 1$ with `future` strategy *after* augmenting the samples. For the `No Augm.` we also relabeled goals with `random` strategy with $p = 0.25$. We train each method for 120k gradient steps, although all methods reach convergence after 90k gradient steps. For all baselines, the percentage of counterfactuals in each batch is set to 0.9 unless specified.

A.2. Experimental details

A.2.1. Franka-Kitchen

We use the kitchen environment from the D4RL benchmark (Fu et al., 2020) which was originally published by Gupta et al. (2019). The D4RL dataset contains different dataset versions: `kitchen-complete, kitchen-partial, kitchen-mixed`, which contain 3690, 136950 and 136950 samples respectively, making up to approximately 14 demonstrations for `kitchen-complete` and 400 demonstrations for each `kitchen-partial` and `kitchen-mixed`. The simulation starts with all of the joint position actuators of the Franka robot set to zero. The doors of the microwave and cabinets are closed, the burners turned off, and the light switch also off. The kettle will be placed in the bottom left burner. The observation are 51-dimensional, containing the joint positions of the robot (9 dim), the positions of the all the kitchen items (21 dim) and the goal positions of all the items (21 dim). The length of the episode is 280 steps, but the episode will finish earlier if the task is completed. The task is only considered solved when all the objects are within a norm threshold of 0.3 with respect to the goal configuration. While in the standard benchmark the agent is required to execute a single fixed sequence of tasks, we train a goal-conditioned agent, and evaluate on one task per each evaluation episode. For the Franka-Kitchen motivating example (see 6.1.1) we query for either the kettle or the microwave task, in the Franka-Kitchen: All tasks (see 6.1.1) we query for the full range of tasks, which include the microwave, the kettle, the slider, the hinge cabinet, the light switch and the bottom left burner tasks. While alleviating the need for long-horizon planning, this results in a challenging setting, as only a subset of tasks is shown directly from the initial configuration. Specifically out of the 1200 demonstrations in the dataset, containing different task sequences, only 3 objects are shown to be manipulated from the initial robot configuration: 60% of the trajectories solve the microwave task first, 30% show the kettle task first and 10% show the bottom burner first. This aligns with the relative performance achieved for those tasks. For the 3 remaining tasks, namely the slide cabinet, the light and the hinge cabinet, there is no demonstration shown directly from the initial configuration and hence the low performance.

Franka-Kitchen: Motivating Experiment For the first experiment (see Subsection 6.1.1), we modify the dataset version `kitchen-mixed` to only contain ~ 50 demonstrations of length ~ 40 timesteps for each `(mw)` and `(k)` task. During demonstrations for the `(mw)` task, we initialize the cabinet to be always open, whereas for demonstrations for the `(k)` task, it remains closed. The rest of the objects are set to the default initial configuration. The goal configuration for all the objects was set to their initial configuration (as defined above), except for the microwave or the kettle, which were set to the default goal configuration when querying for the `(mw)` and `(k)` tasks respectively.

Franka-Kitchen: All Tasks For the second experiment (see Subsection 6.1.1) we merge the 3 provided datasets `kitchen-complete, kitchen-partial, kitchen-mixed`. For this experiment, each object (except the one related to the task at hand to ensure non-trivial completion), was randomly initialized with $p = 0.5$, otherwise it was initialized to the default initial configuration (as defined above). We then modify the desired goal to match the initial configuration for all non-target entities.

A.2.2. Fetch-Push with 2 cubes

Expert data for the experiment in Subsection 6.2.1 includes 6000 episodes collected by an agent trained online using TD3 and HER up to approximately 95% success rate. We additionally collect 14000 episodes with a random agent, which make up for the random dataset. This sums up to a total of 20000 episodes (each of length 100 timesteps), with 30% expert data and 70% random data. Initial positions and goal positions of the cubes are sampled randomly on the table, whereas the robot is initialized in the center of the table with some additional initial random noise . The rewards are sparse, giving a reward of -1 for all timesteps, except a reward of 0 when the position of each of the 2 blocks are within a $2-$norm threshold of 0.05. The observation space is 34-dimensional, containing the position and velocity of the end effector (6dim), of the gripper (4dim) and the object pose, linear and rotational velocities of the objects (12dim each). In contrast to the original `Fetch-Push-v1` (Plappert et al., 2018) environment and similarly to Pitis et al. (2020) we do not include parts of the state space accounting for relative position or velocities of the object with respect to the gripper, which would entangle the two. The goal is 6-dimensional encoding the position for each of the objects. The action space is 4-dimensional encoding for the end-effector position and griper state. At test time, we count the episode as successful upon reaching the goal configuration (i.e., observing a non-negative reward).

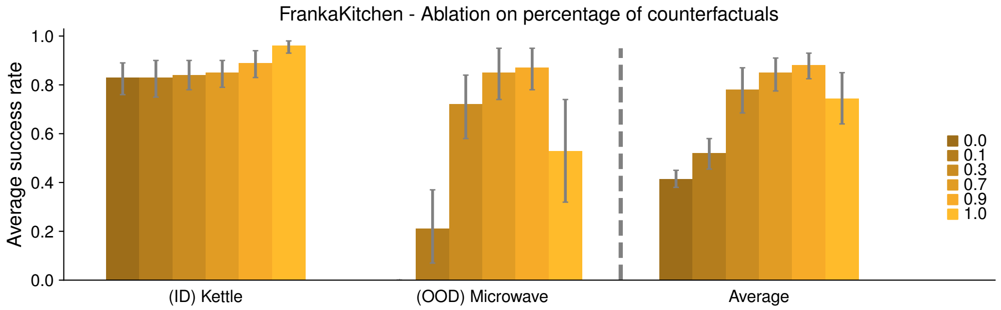


Figure 6: Performance of CAIAC on motivating Franka-Kitchen example when controlling the percentage of counterfactual samples in each batch. Metrics are averaged over 10 seeds and 10 episodes per task, with 95% simple bootstrap confidence intervals.

A.2.3. Fetch-Pick&Lift with 4 cubes

Expert data for the experiment in Subsection 6.1.2 includes 20000 episodes collected by an agent trained online using TD3 and HER up to approximately 95% success rate. We additionally collect 20000 episodes with a random agent, which make up for the random dataset. This sums up to a total of 40000 episodes (each of length 50 timesteps), with same percentage of expert and random data. During data collection, we initialise all the 4 cubes aligned (i.e. all having the same x-axis position). In each episode, we initialise this value from a categorical distribution with 5 options. During testing, the initialization process is the same, but it is done independently for each entity. As a result, the cubes are no longer aligned. The robot is initialized in the center of the table with some additional initial random noise . The rewards are sparse, giving a reward of -1 for all timesteps, except a reward of 0 when the position of each all the cubes are within a $2-$norm threshold of 0.05. The observation space is 58-dimensional, containing the position and velocity of the end effector (6-dimensional), of the gripper (4-dimensional) and the object pose, linear and rotational velocities of the objects (12 dimensions each). In contrast to the original `Fetch-Push-v1` (Plappert et al., 2018) environment and similarly to Pitis et al. (2020) we do not include parts of the state space accounting for relative position or velocities of the object with respect to the gripper, which would entangle the two. The goal is 12-dimensional encoding the position for each of the objects. The action space is 4-dimensional encoding for the end-effector position and griper state. At test time, we count the episode as successful upon reaching the goal configuration (i.e., observing a non-negative reward).

A.3. Ablation: Ratio of observed-to-counterfactual Data

In this section, we study the effect of the ratio of observed-to-counterfactual data generated with CAIAC, by evaluating downstream performance on the Franka-Kitchen motivating example, as presented in 6.1.1. Empirical results for this ablation are shown in Fig. 6. As expected, we observe that the ratio of counterfactuals does not have any significant impact on the success rate on the `(k)` task. This is because the task is evaluated in distribution, and hence the downstream learning algorithm does not require observing counterfactual experience (but still does not suffer from it). For the OOD `(mw)` task we see that increasing the number of counterfactuals up to a 0.9 ratio has a positive effect in performance, leading the agent to generalize better to the OOD distribution. However, when the ratio is increased up to 1, we only use synthesized counterfactual data. We observe a decrease in performance with high variance among training seeds. This hard ablation shows the need for real data during training to avoid induced selection bias, as also observed in (Pitis et al., 2020). With a ratio 0.0, we recover the performance of the No Augm. baseline.

A.4. Details on Influence Detection Evaluation

To detect causal action influence we use CAI, as described in 4.1.

$$C^j(s) := I(S'_j; A \mid S = s) = \mathbb{E}_{a\sim\pi}\left[D_{KL}\left(P_{S'_j|s,a} \;\middle|\middle|\; P_{S'_j|s}\right)\right]. \tag{3}$$

This requires learning the transition model $P_{S'_j|s,a}$.

In the case of robotic manipulation environments *physical contact* is not needed for causal action influence as long as the agent can change the object pose, even if indirectly, in a single simulation step.

World model training For the Franka-Kitchen experiments all models were trained to predict the full state of the environment. For increased performance in the Fetch-Push task, all models were trained to predict the next position of the end effector of the agent gripper and of the objects (3 dimensions each). For CAIAC the transition model $P_{S'_j|s,a}$ is modeled as a Gaussian neural network (predicting mean and variance) that is fitted to the training data $\mathcal{D}$ using negative log likelihood. We used a simple multi-layer perceptron (MLP) with two separate output layers for mean and variance. To constrain the variance to positive range, the variance output of the MLP is processed by a softplus function (given by $\log(1 + exp(x))$), and a small positive constant of 10^{-8} was added to prevent instabilities near zero. We also clip the variance to a maximum value of 200. For weight initialization, orthogonal initialization is used. For the Franka-Kitchen we use larger MLPS, with 3 layers for the simplified and 4 layers for the full experiment, each with 256 units and a learning rate of $8e^{-4}$.

For `CODA` and `CODA-action` we use a self-implementation of the transformer model. We use a model with 3 layers and 4 attention heads for the Fetch-Push task and 5 layers and 4 heads for all the Franka-Kitchen tasks, with an embedding space and output space of 128 dimensions each. We also used a learning rate of $8e^{-4}$.

All models were trained for 100k gradient steps, and tested to reach low MSE error for the predictions in the validation set (train-validation split of 0.9-0.1). We trained all models using the Adam optimizer (Kingma & Ba, 2014), with default hyperparameters.

In general, the models were trained using the same data as for the downstream task for all experiments. However, for the Franka-Kitchen task, we add some additional collected data on the environment when acting with random actions. The reason is that, in order to compute CAI, we query the model on randomly sampled actions from the action space. Due to the expert nature of the kitchen dataset comes from an informed agent, the original dataset might lack random samples and hence we would query the model OOD when computing CAI. For both experiments in the Franka-Kitchen simplified experiment we added 1x the original dataset of random data. Further experiments on the impact of the amount of random data could be beneficial. This was not needed for the Fetch-Push task since the dataset already contains random action. We note that this additional data is also provided for training all transformer-based baselines.

CAI scores In practice, we compute the CAI scores using the estimator:

$$C^j(s) = \frac{1}{K}\sum_{i=1}^{K}[D_{KL}\big(p(s'_j \mid s, a^{(i)}||\frac{1}{K}\sum_{k=1}^{K}p(s'_j \mid s, a^{(k)})\big)] \tag{4}$$

with $a \sim \pi$, where $\pi(A) := \mathcal{U}(\mathcal{A})$ (i.e. a uniform distribution over the action space) and with $K = 64$ actions. We refer the reader to (Seitzer et al., 2021) for more details. In Fig. 7 we show the computed CAI scores over a demonstration in the Franka-Kitchen and in Fig. 8 (left) over an episode for the Fetch-Push task. Given the scores we need a threshold θ to get a classification of control (see Equation 2). For Fetch-Push we set $\theta = 0.05$, for Fetch-Pick&Lift we set $\theta = 2.0$ and for Franka-Kitchen $\theta = 0.2$. See section A.5 for a thorough analysis on the impact of the influence threshold θ and how the parameter was chosen for each of the methods.

Transformer scores To compute causal influence, baselines use the attention weights of a Transformer, where the score is computed as follows. Letting A_i denote the attention matrix of the i'th of N layers, the total attention matrix is computed as $\prod_{i=1}^{N} A_i$. For `CoDA` the score is computed by checking the corresponding row i and column j for the check $s_i \to s'_j$, whereas for `CoDA-action` we restrict ourselves to the row corresponding to the input position of the action component, and the output position of the object component. Our implementation follows Pitis et al. (2020), to which we refer for more details. In Fig. 8 (right) and Fig. 9, we show respectively the computed CoDA-action and CoDA scores over an episode for the Fetch-Push task. Given the scores we need a threshold θ to get a classification of control. For Fetch-Push we set $\theta = 0.2$ for `CODA` and `CODA-action`, for Fetch-Pick&Lift we set $\theta = 0.2$ for `CODA-action` and $\theta = 0.15$ for `CODA`. For Franka-Kitchen we set $\theta = 0.3$ for all methods. See section A.5 for a thorough analysis on the impact of the influence threshold θ and how the parameter was chosen for each of the methods.

A.5. Analysis on Influence Threshold

To get a classification of control, we optimize the value for the threshold θ for all methods. We train 10 different world models for each method and we run a grid search over the parameter θ. We run 3 seeds for each of the 10 models and we picked the value for θ that optimizes the downstream task average performance among the 10 models and the 3 seeds. In Figure Fig. 10 we plot the ROC curves for correct action influence detection for both CAIAC and `CoDA` for the Fetch-Push task. In such an environment, one can specify a heuristic of influence using domain knowledge, namely the agent does not

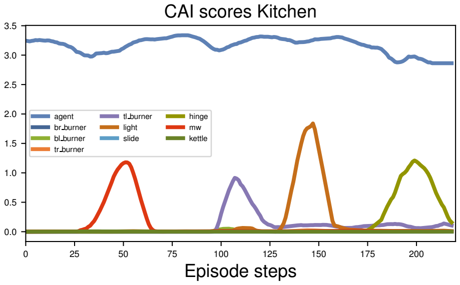


Figure 7: Computed CAI scores per each entity on one of the demonstrations for the Franka-Kitchen dataset. We can observe how the influence of the agent's actions over objects changes over time. First influencing the microwave (mw), then the top-left bottom burner (tl_burner), then the light switch (light) and finally the hinge cabinet (hinge). We selected a threshold of $\theta = 0.3$.

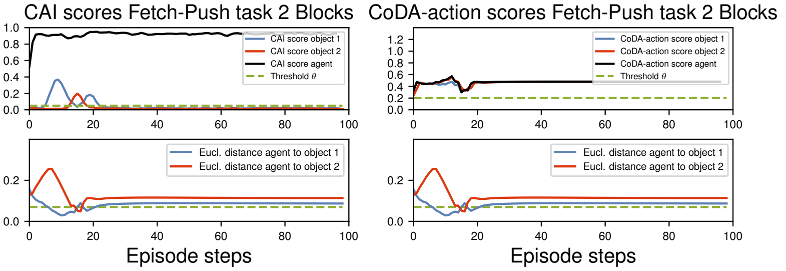


Figure 8: Top left: Computed CAI scores per object on one of the expert demonstrations for the Fetch-Push dataset. We can observe how the influence of the agent's actions over objects changes over time. First pushing object 1, then object 2 and object 1 again. In green we show the optimized threshold $\theta = 0.05$ for this task. Bottom left and right: We show distance from the robot end-effector to each of the objects as a domain knowledge heuristic for action influence. In green we show the heuristic distance of 7cm that we use as a threshold to consider the agent can influence the object within the next timestep. Top right: Computed CoDA-action scores on the same episode as above. In green we show the optimized threshold $\theta = 0.2$ for this task.

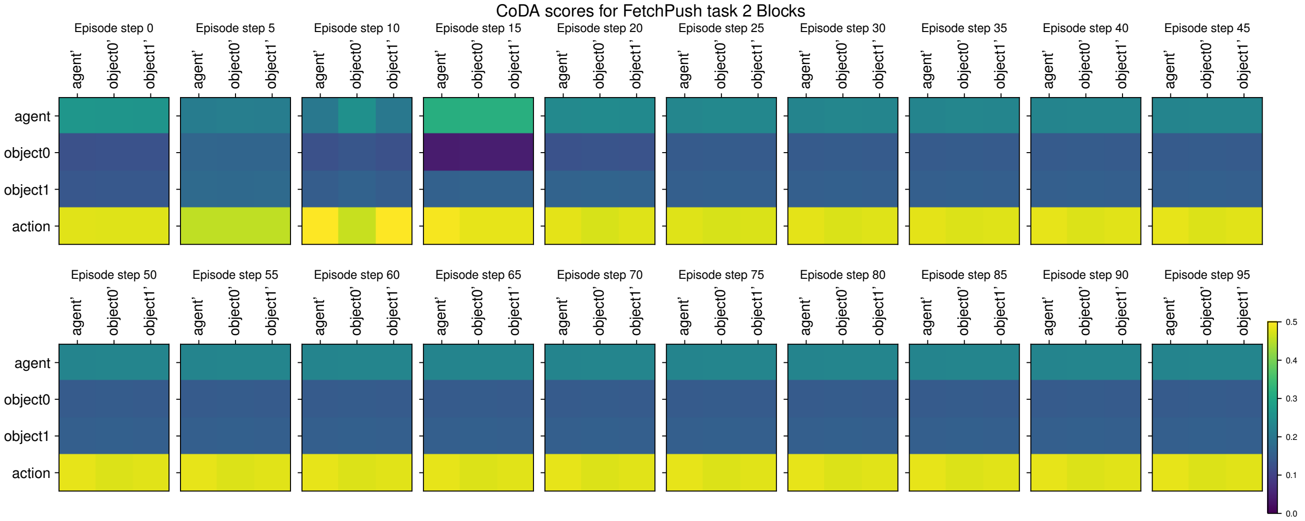


Figure 9: Computed CoDA scores on the same episode as in Fig. 8. We show snapshots of the attention weights of the transformer every 5 time steps as computed by the CoDA algorithm. The element i, j in the matrix shows the attention (or influence) $s_i \rightarrow s'_j$. We see how the transformer completely fails on discovering the full causal graph, being even unable to recover influence along the diagonal, i.e., that an entity state at time t influences the entity state at time t+1.

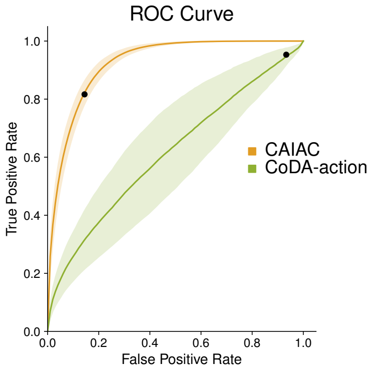


Figure 10: ROC curves for CAIAC and `CoDA-action` (averaged over 10 trained world models and 1 standard deviation shaded). We show measures true and false positive rates (TPR and FPR) while sweeping the influence threshold θ. In black we show the corresponding TPR and FPR for the optimal θ for both methods. See also Figs. 11, 12.

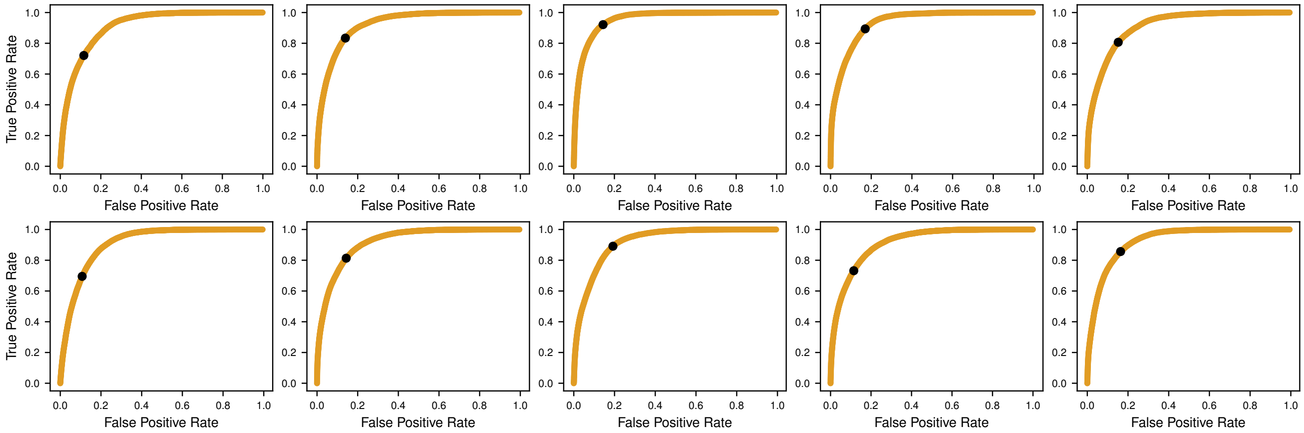


Figure 11: ROC curves for all the 10 trained world using CAIAC. In black we show the corresponding true positive and false positive rate for the optimal threshold $\theta = 0.05$.

have influence on the object if 7cm apart. An accurate model generates an Area Under the Curve (AUC) close to 1, while a random model stays along the diagonal. In Fig. 10 we can observe that the attention weights of the transformer world model are not accurate for detecting influence. Additionally, there is a high variability on the different trained transformers (see also Fig. 12) , making it hard to optimize for the threshold θ for this type of model architecture. In contrast, we see that ROC curves for CAI have an AUC ≈ 0.9 and hence it is an accurate measure for predicting influence in the Fetch-Push environment. Additionally, given its low variance across training seeds (see also Fig. 11), same thresholds reach the same TPR/FPR across models, making it easy to optimize for θ.

A.6. Computational Demands

CAIAC relies on computing the CAI measure for data augmentation. In turn, CAI can be evaluated for all entities at once, through k forward passes for the k counterfactual actions, which are performed in a batch-wise fashion. k is a constant factor, and does not scale with the number of entities. Methods relying on a transformer world model, like CoDA and CoDA-action only need one forward pass (which internally has quadratic cost in the number of entities due to cross attention). However, CoDA also needs to compute the connected components from the adjacency matrix, which has a quadratic cost. For relatively few entities, as is common in the robotic manipulation environments, the computational overhead is relatively small. Table 2 reports an evaluation for the high data regime in the Fetch-Push environment, in which each method was timed while computing influence on all 2M datapoints. The algorithms were benchmarked on a 12-core Intel i7 CPU. We note that counterfactuals could be generated in parallel to the learning algorithm and hence not significantly impact runtime of the algorithm. Furthermore, in our offline setting, counterfactuals can be fully precomputed.

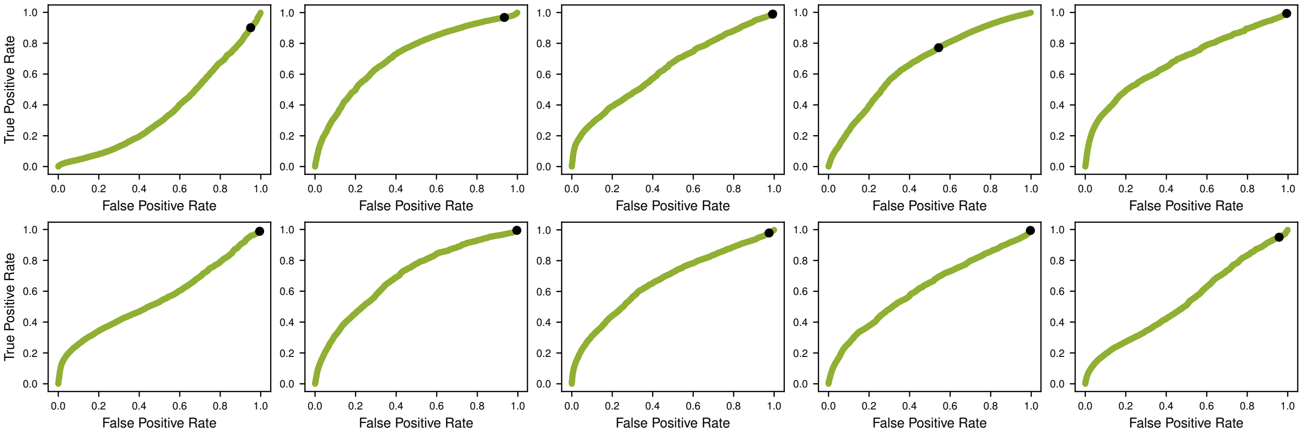


Figure 12: ROC curves for all the 10 trained world using `CoDA-action`. In black we show the corresponding true positive and false positive rate for the optimal threshold $\theta = 0.2$.

A.7. Analysis on the Quality of Created Counterfactuals

We provide an analysis on the created counterfactuals using CAIAC and several baselines, which investigates whether each method (i) creates *feasible* samples (i.e. in accordance to the true transition kernel of the environment) and (ii) increases the support of the joint state space distribution in the training data.

Feasibility The procedure estimating whether the augmentation procedure is valid is as follows. A set of N trajectories $(s_t, a_t, \dots, a_{t+\tau-1}, s_{t+\tau})$ sampled from the Franka-Kitchen dataset is considered. Counterfactuals trajectories $(\tilde{s}_t, a_t, \dots, a_{t+\tau-1}, \tilde{s}_{t+\tau})$ are created for each original trajectory using each method. We then leverage access to the environment's simulator, reset it to the initial counterfactual state $\tilde{s}_t$, and act on the environment by apply actions $a_t, \dots, a_{t+\tau-1}$. If the counterfactual is valid, the resulting state of the simulation $s^{SIM}_{t+\tau}$ should coincide with the counterfactual $\tilde{s}_{t+\tau}$. To account for non-determinism in the simulator, each action sequence is simulated $K = 50$ times, which results in a set of K final states $\mathcal{S}' = \{\bar{s}^k_{t+\tau}\}^K_{k=1}$. A multivariate Gaussian distribution is then fit to the samples from $\mathcal{S}'$ via Maximum Likelihood Estimation.

This allows the computation of the likelihood of the final counterfactual state $\tilde{s}_{t+\tau}$ under the Gaussian distribution for each method, and for each of the N initial trajectories. Fig. 2 presents the density of log-likelihoods for each of the methods, approximated with a Gaussian KDE. We observe that CAIAC's augmented data have high likelihood under the distribution of final states returned by the simulator, and hence are mostly valid. In contrast, while some counterfactual trajectories generated by other methods are also viable, most of their samples are associated to low log-likelihoods, which suggests a violation of the environment's dynamics.

Increased support Fig. 2 (right) shows 1000 randomly selected samples from the Franka-Kitchen dataset, as well as one counterfactual (created using CAIAC) for each. The visualization employs t-SNE for a 2D representation of the high-dimensional state space. We observe how the augmented samples cover a larger space than the observed data, suggesting that the support of the joint training distribution over entities is improved. We also provide numerical evidence by comparing the support of the empirical distributions. For each sample augmented with all methods, we first bin each of the object states into 2 categories. We then compute the support of the resulting categorical distribution with 2^N possible categories, where N is the number of objects. Table 3 reports ratios between the estimated support and the theoretical maximum.

Table 2: Computational demands for computing counterfactuals for the different algorithms. Runtime was benchmarked on a 12-core Intel i7 CPU.

	CAIAC	CoDA	CoDA-action
Runtime (min)	$\sim$13	$\sim$10	$\sim$1

In 3 we show how CAIAC increases the support of the original state space distribution by almost a factor of 2 . However, we can see that rest of the baselines also increase it significantly. Importantly, this numbers do not inform on whether these counterfactuals are actually valid, i.e. follow the true transition kernel of the environment, which was already investigated in the above Feasibility paragraph.

B. Implementation Details for Baselines

In the following we provide implementation details for `RSC` and additional model-based baselines. Details regarding CAIAC and `CoDA` and `CoDA-action` baselines were already provided in the respective sections above.

B.1. RSC (Ding et al., 2023)

Due to lack of public code available by the time of the submission, we resort to reimplementing RSC (Ding et al., 2023). The implementation follows the guidelines described in the original paper, with a few differences. In the original method, the dynamics model leverages a learned causal graph as a gating mechanism to achieve better generalization to perturbed state and action pairs; training is regularized to encourage sparsity in this graph. In our case, we instead leverage a transformer dynamics model, in which sparsity is granted by the softmax in its attention mechanism. Additionally, while the dynamics model in RSC is trained to predict both rewards and next states, in our case predicting the next state is sufficient, as the ground truth reward can be computed with the relabeling function available in goal-conditioned settings.

A final difference that ensures a fair comparison is in the perturbation phase. The original description of RSC is not object-centric, and thus only perturbs a single dimension of the state space. In our experiments, we ensure that RSC also leverages the known decomposition of the state space in objects, and thus augment the state of a single object in a consistent manner.

Finally, a naive maximization of the heuristic (Equation 7 in Ding et al. (2023))over the entire dataset $\mathcal{D}$ and K objects would require evaluating $K|\mathcal{D}|^2$ distances. Due to compute constraints, we optimize it by subsampling 1000 candidate solutions, which we found to still approximate the maximum reasonably well.

B.2. Model based baselines

We present a performance comparison between CAIAC and an MBPO-style causal-unaware model-based approach, which we call `MBPO` for simplicity.

`MBPO` leverages a dynamical model trained from data to generate on-policy imagined N-step transition rollouts. Despite the ease of data generation, these methods often suffer from bias of model-generated data, which leads to approximation errors that compound with increasing the horizon (N).

In our spurious correlation experiments, we would argue that it is hard for the model to accurately autoregressively predict long horizons trajectories, that would lead the agent to OOD states useful to solve the task at hand. Additionally, if that would be the case, then a causal-agnostic model would be queried OOD suffering leading to poor generalization. Work on the direction for task independent state abstraction tackles this issue (Wang et al., 2022).

We show the results for the reinforcement learning experiments (namely FetchPush and Fetch-Pick&Lift) in Fig. 13.

For the spurious correlation environment Fig. 13 (left), we indeed confirm our hypothesis, were `MBPO` is unable to create samples that break the spurious correlation in the data, and hence leads to poor performance compared to CAIAC.

However, in the low data regime environments Fig. 13 (right), we observe how `MBPO` thrives and outperforms CAIAC. We hypothesize that in this environment, the trained model is accurate enough even in low data regimes, and hence can generate useful trajectory rollouts that increase the amount of data substantially. Since spurious correlations are not present in the

Table 3: Support of the empirical joint state space distribution using different augmentation methods (results computed over 1000 augmented samples).Values represent ratios between the estimated support and the theoretical maximum.

	CAIAC	CoDA	CoDA-action	No Augm.
Support	0.52	0.48	0.42	0.3

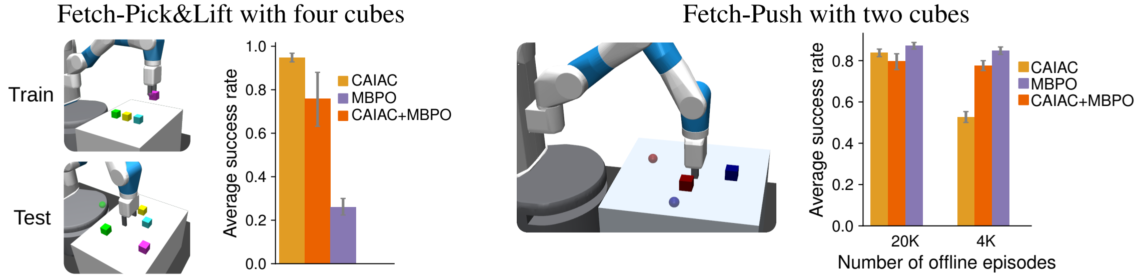
data, the model doesn't need to be queried OOD to create meaningful samples.

We finally decide to leverage the strengths of both approaches and propose `CAIAC+MBPO`. With this approach, we train a dynamics model both *on the original AND* CAIAC*'s augmented data*. Notably, the model is trained on perturbed yet dynamically feasible samples, potentially leading it to better generalization capabilities in OOD states. Finally, we we train our agent with `MBPO` using both the *augmented* CAIAC *and the original samples*.

We observe that in low data regimes Fig. 13 (right), CAIAC+MBPO leads to a boost in performance compared to CAIAC alone. However, it still underperforms with respect to `MBPO`. We hypothesize that this is due to some fraction of unfeasible augmented samples used to train the model, leading to compounding errors when generating the rollout transitions with `MBPO`. This would also support the results for the Fetch-Pick&Lift environment Fig. 13(left), where despite outperforming `MBPO` significantly, `CAIAC+MBPO` doesn't achieve same performance as `CAIAC` alone.

In terms of implementation details, all the methods share the same hyperparameters, except for the horizon length (N) and the ratio of augmented samples (R) which we tuned individually. We used $N = 5$ for `CAIAC+MBPO` and $N = 10$ for `MBPO` and $R = 0.5$ for the FetchPush environment, while for FetchPick&Lift we crucially reduce R to 0.1 for all methods and use $N = 5$.

We believe that combining CAIAC with model-based approaches is a very promising direction, and we leave extensive exploration of such an approach for future work.


Fetch-Pick&Lift with four cubes

Fetch-Push with two cubes

Figure 13: Success rates for CAIAC and model-based approaches in Fetch-Pick&Lift with 4 objects (left) and Fetch-Push with 2 cubes (right). Metrics are averaged over 30 seeds and 50 episodes with 95% simple bootstrap confidence intervals.