Causal Transfer for Imitation Learning and Decision Making under Sensor-shift

Jala Etesami, Philipp Geiger
Bosch Center for Artificial Intelligence - BCAI
Robert Bosch GmbH
71272 Renningen, Germany
Jalal.Etesami@de.bosch.com,
Philipp.W.Geiger@de.bosch.com

Abstract

Learning from demonstrations (LfD) is an efficient paradigm to train AI agents. But major issues arise when there are differences between (a) the demonstrator's own sensory input, (b) our sensors that observe the demonstrator and (c) the sensory input of the agent we train.

In this paper, we propose a causal model-based framework for transfer learning under such "sensor-shifts", for two common LfD tasks: (1) inferring the effect of the demonstrator's actions and (2) imitation learning. First we rigorously analyze, on the population-level, to what extent the relevant underlying mechanisms (the action effects and the demonstrator policy) can be identified and transferred from the available observations together with prior knowledge of sensor characteristics. And we device an algorithm to infer these mechanisms. Then we introduce several proxy methods which are easier to calculate, estimate from finite data and interpret than the exact solutions, alongside theoretical bounds on their closeness to the exact ones. We validate our two main methods on simulated and semi-real world data.

1 Introduction

Motivation. Learning from demonstrations is an important paradigm to train AI agents (**?**; **?**; **?**; **?**). Ideally, one would like to harness as much *cheaply available (and relevant) demonstrator data* as possible. But major issues arise when there are *differences between the sensors* of demonstrator, us and agent we train. When ignoring such issues, or addressing them in a naive way, wrong and potentially harmful conclusions can result: about demonstrator's behavior and the demonstrator's actions' effects on the environment.

Example 1 (Highway drone data)**.** In the development of self-driving cars, recently drones have been deployed to fly over highways and record the behavior of human-driven cars (**?**; **?**). Clearly, in such drone recordings, some crucial variables are either *more noisy* than observed from within the car, or completely *missing*, such as *indicator lights*.

Assume we want to use such data to learn, say, how an acceleration action A of a "demonstrator car" affects the lane changing behavior Z of a "lead car" in front of it on the slower lane, as depicted in Figure 1. Slightly simplifying reality, assume the indicator light of the lead car serves as a perfect coordination device: it is on if and only if, subsequently, (1) the demonstrator car decelerates and (2) the lead car changes lane to the fast lane. Now assume we just use the variables recorded in the drone data, where the indicator light is not contained, estimate $P(Z|A)$ from it, and naively consider it as the *causal effect of A on Z*.

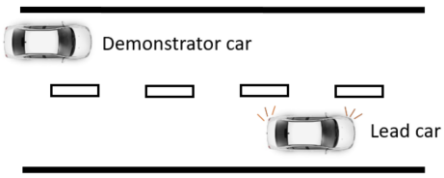

Figure 1: In highway drone data, the *indicator light* of the lead car would be missing, introducing a *hidden common cause* between acceleration of demonstrator car and lane changing behavior of the lead car.

This leads us to the conclusion that an agent in the place of the demonstrator can arbitrarily chose any acceleration or deceleration action as A, and the lead car will perfectly adapt Z and only change lane when agent decelerates – which in practice can lead to crashes. In the language of causal models (**?**; **?**), the indicator light is a *hidden common cause (confounder)*.

Main tasks, approach and contributions: In this paper, we address learning from demonstrations (LfD) under *sensor-shift*, i.e., when there are differences between (a) the demonstrator's own sensory input, (b) our sensors that observe the demonstrator and (c) the sensory input of the agent we train. Specifically, we consider two closely related "subtasks" of LfD: (1) inferring the effect of the demonstrator's decisions (as in Example 1) and (2) imitating the demonstrator.

Our approach is based on causal models (**?**; **?**; **?**), which allow us to generalize from data beyond i.i.d. settings. The idea is that, while some modular causal mechanisms that govern the data vary (the sensors), other mechanisms are *invariant* (e.g., the action-effect).

Our main contributions are:

- We rigorously analyze, on the population-level, to what extent the relevant underlying mechanisms (the action-effect and the demonstrator policy) can be *identified* and transferred from the available observations together with prior knowledge of sensor characteristics (Sections 5, 6.1, 6.2 and 7.1). And we propose algorithms to calculate them (Algorithms 1 and 2).
- We introduce several *proxy methods* (Sections 6.3 and 7.2) which are easier to calculate, estimate from finite data and interpret than the exact solutions, alongside theoretical bounds on their closeness to the exact ones (Propositions 2, 4 and 5). (Proofs are in the supplement[1] of this paper.)
- We conduct *experiments* to validate our two main methods on *simulated and semi-real world* highway drone data used for autonomous driving (Section 8).

2 Related work

Learning from demonstrations (LfD) (**?**) is a broad area, with two concrete tasks being the ones we also consider in this paper: (1) inferring the effect of action on outcome given observation (we call it "action-effect" in our a-temporal framework, while in the language of (**?**) this is called the "system model" or "world dynamics"), and (2) imitation learning (see next paragraph). Generally in LfD, the problem that sensors differ between demonstrator, observer and target AI agent has been considered (**?**; **?**; **?**). In the language of (**?**), this is described as the "recording mapping" or "embodiment mapping" not being the identity. However, we are not aware of any treatment of this problem which is as *systematic and general* as ours in terms of guarantees on exact and approximate identifiability. Instead, approaches are practically-focused, tailored to specific, say, robot tasks (**?**; **?**).

Within LfD, *imitation learning* means learning to perform a task from expert demonstrations (**?**; **?**). There are two main approaches to address this problem: behavioral cloning (**?**), which we are focusing on, and inverse reinforcement learning (IRL) (**?**; **?**).

The problem of bounding as well as transferring and integrating *causal relations* across different domains has been studied by (**?**; **?**; **?**). But all this work does not consider the training of AI agents. Within causal modelling, maybe closest related to our paper are (**?**; **?**; **?**; **?**), who also study the integration of data from heterogeneous settings for training agents (often with latent confounders and from a multi-armed bandit perspective).

For example, (**?**) tackle the problem of transferring knowledge across bandit agents in settings where causal effects cannot be identified by standard learning techniques. Their approach consists of two steps: (1) deriving bounds over the effects of selecting arms and (2) incorporating these bounds to search for more promising actions. However, when bounding the causal effect, they focus on binary variables, while we consider arbitrary finite as well as continuous ranges (which are highly relevant in practice) and they do not focus on general sensor-shifts.

The authors of (**?**) study "causal confusion" in causal-model-free imitation learning. There, additional observations can lead to worse performance due to the mechanism (policy) that generates them differing between demonstrations and target environment. However, in their model they assume that both the demonstrator and the imitator have (at least) the same observations. This is not always the case, and therefore our treatment allows the observations to differ.

3 Background

Conventions: We use $D(\cdot||\cdot)$, $H(\cdot)$, and $I(\cdot;\cdot|\cdot)$ to denote the Kullback-Leibler (KL) divergence, entropy, and mutual information, respectively (**?**). We consider both, discrete and continuous random variables; $\nsum$ stands for the sum or integral, accordingly; $P(W)$ for the distribution of a variable W, and $p(w)$ for the density at value $W = w$. If not stated otherwise, we assume that distributions have full support[2] and densities.

Causal models: According to Pearl's definition (**?**), a *causal model* is an ordered triple (U, V, E), where U denotes a set of *exogenous variables* whose values are determined by factors outside the model (not observable); V is a set of *endogenous variables* whose values are determined within the model; and E is a set of *structural equations* that express, for each endogenous variable $W \in V$, the *mechanism* of how W is generated by certain other endogenous and exogenous variables. Namely, for all $W \in V$, we have

$$W = f_W(PA_W, U_W),$$

where $f_W(\cdot,\cdot)$ is a function and PA_W denotes the *parent set* of variable W. W is called a *child* of PA_W. This induces a joint distribution over the endogenous variables, which can be factorized as follows:

$$P(V) = \prod_{W \in V} P(W|PA_W).$$

This factorization is usually expressed using a *directed acyclic graph (DAG)*, in which nodes represent the endogenous variables and arrows are from parents to their children. It is also possible that a sub-set of V is hidden. In this case, we denote the hidden variable with circles in the DAG.

The *post-interventional distribution* is defined by replacing a subset of structural equations without generating cycles in the DAG (**?**). More specifically, the post-intervention distribution after (*atomic*) intervening on variable W is defined by replacing $f_W(PA_W, U_W)$ with value w and it is denoted by $P\big(V|do(W = w)\big)$.

4 Setting and problem formulation

4.1 General model of our setting

Causal models of source and target domain. There are two domains, the *source domain* where the *demonstrator*

[1]The supplement can be found at "https://doi.org/10.5281/zenodo.3549981".

[2]Full support is a commonly made (**?**) but non-trivial assumption, important for identifiability.

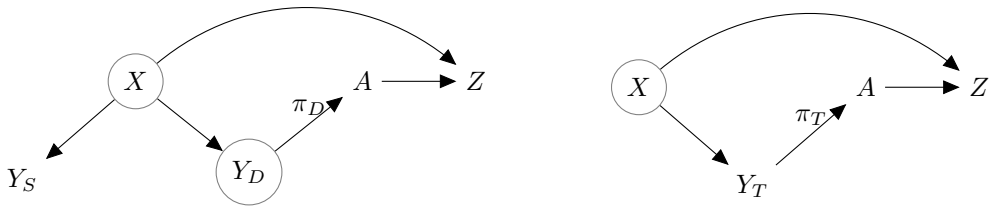

Figure 2: Causal DAGs. **Left:** source domain. **Right:** target domain. Circle means hidden to us.

(agent) observes and acts, and the *target domain* where the *target agent*, which we design, observes and acts. (By domain we mean the complete causal model of environment, sensors, and agent.) The two domains, including what is hidden and what is observed by us, are depicted by the two causal DAGs in Figure 2 over the following variables: X is the *state* of the system, A is the *action* of the agent, Z stands for the *outcome* (an abstract variable that could be, as in Example 1, the state of cars in the the next time instance). Regarding *observations*, we assume that in the source domain we have Y_D, the *demonstrator's input*, generated by the demonstrator's sensors, Y_S, the *spectator's – i.e., our – observation* of the state of the source system, and in the target domain we have Y_T, the *input to the target agent* from the target agent's sensors. We often denote distributions over variables (e.g. $P(Z)$) in the source and target domain by subscript S and T, respectively (e.g., $P_S(Z)$ and $P_T(Z)$). Let $\pi_D(A|Y_D)$ denote the *policy of the demonstrator*, and $\pi_T(A|Y_T)$ denote the *policy of the target agent*.

Relationship between source and target domain, and what is known to us. We assume that the two domains are related by sharing the same invariant mechanism for outcome given state and action, i.e.,

$$P_T(Z|A,X) = P_S(Z|A,X),$$

so that we can drop the subscript and just write $P(Z|A,X)$. We assume we are given $P_S(Z,A,Y_S)$ (or a sample of it), as well as the sensor characteristics[3] $P_S(Y_S|X)$ and $P_T(Y_T|X)$.

4.2 Problem formulation

The overarching goal is to design the target agent that observes and successfully acts in the target domain, based on what we know from the source domain and its relation to the target domain. We consider two specific tasks that serve this overarching goal:

Task 1 (Action-effect transfer learning task)**.** Infer $P_T(Z|do(A),Y_T)$, the effect of action A on outcome Z conditional on observation Y_T in the target domain.[4]

[3]This may be based on performing an experimental system identification of the sensors or using physical knowledge.

[4]Once the effect $P_T(Z|Y_T,do(A))$ is inferred, what remains to be done for designing the target agent is to fix a utility function $u(Z)$ on the outcome, and then pick the optimal a by, say, maximizing $\mathbb{E}_T(u(Z)|do(a),y_T)$ w.r.t. a.

Task 2 (Imitation transfer learning task)**.** Learn a policy $\pi_T(A|Y_T)$ for the target agent (also called *imitator* in this task) such that it behaves as similarly as possible to the demonstrator (details follow).

5 Basic step addressing both tasks: equations and algorithm

In this section, we make general derivations about our model (Section 4.1), which serve as steps towards *both*, the imitation and the action-effect transfer learning tasks.

Basic equation: Our model (Section 4.1) implies the following equations, for all z, a, y:

$$p_S(z,a,y_S) = \sum\!\!\!\!\!\!\int_x p_S(y_S|x)p_S(z,a,x) \tag{1}$$

$$= \sum\!\!\!\!\!\!\int_{x,y_D} p_S(y_S|x)p(z|a,x)\pi_D(a|y_D)p_S(y_D,x). \tag{2}$$

These are the basic equations that relates what is known – $p_S(z,a,y_S)$ (l.h.s. of (1)) – to what we would like to know (r.h.s. of (2)): $\pi_D(a|y_D)$ for Task 2 and $p(z|a,x)$ for Task 1. More specifically, these equations *constrain* the unknown quantities to a set of possibilities. This is exactly the set up to which we can *identify* (**?**) them.

Finite linear equation system in discrete case: Solving (1) for $p_S(z,a,x)$ is an important intermediate step to addresses Task 1 and 2 simultaneously, since $p_S(z,a,x)$ contains all the information that $p_S(z,a,y_S)$ contains about $\pi_D(a|y_D)$ and $p(z|a,y_T)$. (In particular, in the classical case of $Y_S = Y_T = Y_D = X$, $p_S(z,a,x)$ uniquely determines the latter two quantities via marginalization/conditioning.) So let us for a moment focus on (1). In the discrete case, it can be rewritten as the following collection of matrix equations. Let $\{x^1,\dots,x^\ell\}$ and $\{y^1,\dots,y^m\}$ be the range of X and Y_S, respectively. Then, for all z, a,

$$\underbrace{\begin{bmatrix} P(z,a,y^1) \\ \vdots \\ P(z,a,y^m) \end{bmatrix}}_{P(z,a,Y_S)\in\mathbb{R}^m} = \underbrace{\begin{bmatrix} P(y^1|x^1)\cdots P(y^1|x^\ell) \\ \vdots \quad\quad \vdots \\ P(y^m|x^1)\cdots P(y^m|x^\ell) \end{bmatrix}}_{[P(y^i|x^j)]_{i,j=1}^{m,\ell}\in\mathbb{R}^{m\times\ell}} \underbrace{\begin{bmatrix} P(z,a,x^1) \\ \vdots \\ P(z,a,x^\ell) \end{bmatrix}}_{P(z,a,X)\in\mathbb{R}^\ell}. \tag{3}$$

Algorithm 1: Finding solution set for (3)

Input: $P(z, a, Y_S)$ (l.h.s. of (3)), $[P(y^i|x^j)]_{i,j=1}^{m,\ell}$

Output: $\zeta_1, \ldots, \zeta_k \in \mathbb{R}^\ell$, such that their convex hull is the solution set to (3)

1 Rearrange columns of $[P(y^i|x^j)]_{i,j=1}^{m,\ell}$ such that $[P(y^i|x^j)]_{i,j=1}^{m,\ell} = [D\ E]$ and $D \in \mathbb{R}^{m\times m}$ is non-singular;

2 $U\Sigma V^T \leftarrow$ SVD of $[P(y^i|x^j)]_{i,j=1}^{m,\ell}$;

3 **for** $i = 1$ **to** $\ell - m$ **do**

4 $\quad e_i \leftarrow$ zero vector of length $\ell - m$ whose ith entry is one;

5 $M \leftarrow V \begin{bmatrix} \mathbf{0} & \cdots & \mathbf{0} \\ e_1 & \cdots & e_{\ell-m} \end{bmatrix}$, $b \leftarrow \begin{bmatrix} D^{-1}P(z, a, Y_S) \\ \mathbf{0} \end{bmatrix}$;

6 $i \leftarrow 1$;

7 **for** *any sub-matrix R of M with dimension* $(\ell - m) \times (\ell - m)$ **do**

8 $\quad \hat{b} \leftarrow$ the sub-vector of b of length $\ell - m$ that corresponds to the selected rows of M;

9 $\quad$ **if** R^{-1} *exists and* $-MR^{-1}\hat{b} + b \geq 0$ **then**

10 $\quad\quad \zeta_i \leftarrow -MR^{-1}\hat{b} + b$;

11 $\quad\quad i \leftarrow i + 1$;

Algorithm for solution set in discrete case: Algorithm 1 yields a parametrization of the set of all possible solutions $P(z, a, X) \in \mathbb{R}^\ell$ to (3), for any z, a. Specifically, it outputs the finite set of *corner vectors* whose *convex combinations parametrize the solution set*.

It uses singular-value decomposition (SVD) to cope with non-invertibility, and then a routine inspired by the simplex algorithm to account for the constraint that the output has to be a proper probability distributions.[5]

For the algorithm, w.l.o.g., we assume $m \leq \ell$ and that $[P(y^i|x^j)]_{i,j=1}^{m,\ell}$ has full rank (otherwise one removes linearly dependent rows). Note that if $m = \ell$ and $[P(y^i|x^j)]_{i,j=1}^{m,\ell}$ is non-singular, then (3) determines $P(z, a, X)$ uniquely, via a simple matrix inversion. Therefore, for this algorithm, the interesting scenario is $m < \ell$. This is the case, e.g., in Example 1 – the highway drone data where indicator lights are not recorded.

6 Approach to the action-effect transfer learning task

Let us now address Task 1 – inferring the target domain's action-effect $P_T(Z|do(A), Y_T)$.

Example 2. To illustrate what can go wrong when naively addressing this task, let us get back to the highway drone data (Example 1). There, in the source domain, the indicator light is not observed by us, and for simplicity we assumed that there are no other variables, i.e., Y_S is empty/constant. Our informal argument in that example can now be stated formally based on causal models (Section 3): Observe that in the causal DAG (Figure 2), X, Y_D are *hidden confounders* that introduce "spurious correlations" between A and Z. Therefore, in the generic case, the naive guess $P_S(Z|a)$ does not coincide with the actual action-effect $P_S(Z|do(A))$ $(= P_T(Z|do(A)))$.

Assumption 1. In this section, we assume the target agent observes the full state, i.e., $Y_T{=}X$.[6]

Under Assumption 1, we have

$$P_T(Z|do(A), Y_T) = P_T(Z|do(A), X) = P(Z|A, X).$$

So Task 1 means inferring $P(Z|A, X)$ (which could also be referred to as the (target domain's) "dynamics"). We now propose three methods, which differ w.r.t. the setting in which they are applicable and/or w.r.t. yielding exact or approximate solutions.

6.1 Exact solution set in the discrete case

In the case of all variables being discrete, we can build on our basic step in Section 5 to analytically find the set of possible action-effects $P(Z|X, A)$ as follows: first we deploy Algorithm 1 to get all possible $P(Z, X, A)$, and then from this (simply by dividing by the marginals), we get $P(Z|X, A)$.

6.2 Exact solution in the linear invertible continuous case

In the continuous case, the general identification analysis – the analysis of the solution set of (2) – is very difficult because the vectors space is infinite-dimensional. Therefore let us here consider the special case of *linear* relationships.

Assumption 2. In this Section 6.2, assume all relationships are linear, in particular, for matrices D, E, F,

$$Y_S = FX + N, \tag{4}$$

$$Z = [D\ E] \begin{bmatrix} A \\ X \end{bmatrix} + O \tag{5}$$

with N, O the usual noise terms that are independent of all other (non-descendant) variables.

We propose Algorithm 2 as (sample-level) method in this setting.

Proposition 1. Assume all variables have mean zero (otherwise center them). Furthermore, assume that X and Y_S have the same dimension, and that F (in (4)) is invertible. Then Algorithm 2 is *sound* in the following sense: when replacing the empirical covariance matrices

$$\hat{\Sigma}_{ZA}, \hat{\Sigma}_{ZY_S}, \hat{\Sigma}_{AY_S}, \hat{\Sigma}_{Y_SY_S}$$

in Line 1 by their population-level counterparts, and setting the regularization term $\lambda = 0$, the output will be the true D, E (in (5)).

[5] Since the left hand side of (3) is a probability vector, it is not necessary to bound $P(z, a, x^i)$ by one.

[6] Observability of X, similar as in Markov decision processes (MDPs), seems to be a good approximation to many real-world situations while at the same time keeping the analysis instructive. We make no assumption w.r.t. Y_D.

Algorithm 2: Exact linear action-effect transfer method (sample-level)

Input: sample $(z_1, a_1, y_1), \ldots, (z_\ell, a_\ell, y_\ell)$ from $P(Z, A, Y_S)$; prior knowledge F, Σ_{NN} (see (4)); regularization parameter λ

Output: Estimates $\hat{D}, \hat{E}$ for the regression matrices D, E (see (5))

Calculate the empirical covariance matrices $\hat{\Sigma}_{ZA}, \hat{\Sigma}_{ZY_S}, \hat{\Sigma}_{AY_S}, \hat{\Sigma}_{Y_SY_S}$ from the sample

Add a regularization term $\lambda \mathbf{1}$ to $\hat{\Sigma}_{AA}$ and $\hat{\Sigma}_{Y_SY_S}$

Calculate the Schur complements $S_1 := \hat{\Sigma}_{AA} - \hat{\Sigma}_{AY_S}(\Sigma_{Y_SY_S} - \Sigma_{NN})^{-1}\hat{\Sigma}_{Y_SA}$, $S_2 := \hat{\Sigma}_{Y_SY_S} - \Sigma_{NN} - \hat{\Sigma}_{Y_SA}\hat{\Sigma}_{AA}^{-1}\hat{\Sigma}_{AY_S}$.

Calculate the estimates $\hat{D} := \hat{\Sigma}_{ZA}S_1^{-1} - \hat{\Sigma}_{ZY_S}(\hat{\Sigma}_{Y_SY_S} - \Sigma_{NN})^{-1}\hat{\Sigma}_{Y_SA}S_1^{-1}$, and $\hat{E} := -\hat{\Sigma}_{ZA}S_1^{-1}\hat{\Sigma}_{AY_S}(\hat{\Sigma}_{Y_SY_S} - \Sigma_{NN})^{-1} + \hat{\Sigma}_{ZY_S}S_2^{-1})F$

6.3 Average-based action-effect proxy in the general case

The *exact* general solution can be difficult to handle in terms of computation, estimation and analysis, and the linear case (Section 6.2) is of course restrictive. Let us define the following *average-based action-effect proxy* of the density $p(z|x, a)$, for all z, x, a, defined only based on things we do know (from the source domain):

$$\tilde{p}(z|x, a) := \sum_{y_S} p_S(z|y_S, a)p(y_S|x), \tag{6}$$

and let $\tilde{P}(Z|X, A)$ be the corresponding distribution. The deviation between the average-based proxy and the ground truth it approximates can be bounded as follows:

Proposition 2. We have[7]

$$D(P_S(Z|X, A)\|\tilde{P}(Z|X, A)) \leq I_S(X; Z|A, Y_S).$$

In particular, if $Y_S = f_{Y_S}(X)$ with f_{Y_S} injective, then $\tilde{P}(Z|X, A) = P(Z|X, A)$. Note that in the discrete case, the r.h.s. in turn can be bounded by an expression that is solely based on quantities, which we assumed to know: $\max_{P'(X)} H_{X\sim P'(X)}(X|Y_S)$.

7 Approach to the imitation learning task

In this section, we address Task 2. To do so, we propose an imitator (the target agent) that selects a policy $\pi_T(A|Y_T)$ such that its behavior[8] is as close as possible to the demonstrator.

[7] In fact we bound the KL divergence between proxy and $p(Z|X, A)$, but the expectation over X, A is w.r.t. the *source* domain, and therefore we have to write $p_S(Z|X, A)$ on the l.h.s. of $D(\cdot\|\cdot)$. See also the proof.

[8] Our notion of behavior is the conditional distribution of the action-outcome pair given the observation.

Recall that, for the design of the imitator, what is available about the demonstrator is (a sample from) $P_S(Y_S, Z, A)$. However, the challenge is that the observation set of the demonstrator and the imitator may not be the same. Therefore, we propose an imitator that behaves as close as possible to the demonstrator in case of perfect observation, i.e.,

$$\arg\min_{\pi_T} D\Big(P_T(A, Z|X)||P_S(A, Z|X)\Big). \tag{7}$$

It is worth noting that the imitator can also introduce additional constraints to this optimization problem according to its environment. Next, we give a simple example to illustrate what can go wrong when naively addressing the imitation task under sensor-shift. Then we propose methods for the problem in (7) for several settings.

Example 3. Let us come back to Example 1 and Figure 1, where the indicator light perfectly correlates deceleration and lane changing. Let us add some modifications: Assume we have the same sensors to observe the demonstrator as we have on board of the imitator's car, i.e., spectator's and imitator's sensors coincide, $P(Y_T|X) = P(Y_S|X)$. And assume these sensors (similar to the drone) are missing the indicator light of the lead car (unlike the demonstrator's observation Y_D). Now, for the imitation task at hands, assume we naively take $\pi_T(a|y_T) := p_S(a|Y_S = y_T)$ as the imitator's policy.

This means that the imitator will accelerate and decelerate randomly, instead of, as the demonstrator, perfectly adapting these actions to the indicator light of the lead car (the indicator light is the actual source of variation in A given Y_D, but the imitator just takes $P_S(A|Y_S)$ for a randomized policy). This will necessarily lead to crashes in the target domain – whenever the lead car indicates and the imitator randomly decides to accelerate. This issue can also be seen formally, based on the causal DAG (Figure 2): there is a back-door path (**?**) between action A and outcome Z that is not blocked by Y_S, and therefore, in the generic case, $P_S(Z|do(A), Y_S) \neq P_S(Z|A, Y_S)$.

7.1 Exact solution set in the discrete case

Assumption 3. Here we assume that both the demonstrator and the imitator have the same sensors[9], i.e.,

$$P_S(Y_D|X) = P_T(Y_T|X).$$

Proposition 3. Given Assumption 3, the solution of (7) is

$$\pi_T(a|Y_T = y) := \pi_D(a|Y_D = y).$$

Although this result introduces the optimal policy for the imitator, it is practical only if the imitator can infer $\pi_D(a|Y_D = y)$ using its observation from the source domain. In case of all variables being discrete, the imitator is able to do so using a set of finite linear equations similar to Section 5. More precisely, (2) leads to

$$P_S(a, y_S) = \sum_y P_S(y_S|Y_D = y)P_S(a, Y_D = y). \tag{8}$$

[9] However, we relax this assumption in the next section.

Assumption 4. For the rest of Section 7, we assume that $P_S(A, Y_S), P_S(Y_S|Y_D)$ are known to the imitator.

This forms a set of equation similar to (3). Algorithm 1 (with input $P(a, Y_S)$, $[P(y_S^i|y_D^j)]_{i,j=1}^{m,\ell'}$, with ℓ' denoting the size of the range of Y_D) obtains the set of possible $P_S(a, Y_D)$ and consequently

$$\pi_D(a|Y_D) = \frac{P_S(a, Y_D)}{\sum_{a'} P_S(a', Y_D)}.$$

Remark 1. Generally, it is important to mention that such assumptions can be weakened. But it will significantly increase the complexity of the problem by essentially adding another layer of non-unique-identifiability of the joint from the conditional, e.g., $P_S(X, Y_S)$ from $P_S(Y_S|X)$.

7.2 Average-based proxy in the general case

Here, we propose proxy methods, which have the advantage that they can also be applied to the continuous case and may be easier to estimate/compute. We do so for three different cases of sensor-shift.

First case: In this case, the imitator and the demonstrator have the same sensors in their domains, but the other sensors can be different, i.e., $P_T(Y_T|X) = P_S(Y_D|X)$. Based on Proposition 3, the optimal policy for the imitator is indeed π_D. Thus, we propose the following policy: $\tilde{\pi}_T^{(1)}(a|Y_T = y) := \tilde{\pi}_D(a|Y_D = y)$, where the latter is defined by

$$\sum\!\!\!\!\!\!\int_{y'} p_S(a|Y_S = y')p_S(Y_S = y'|Y_D = y). \tag{9}$$

Proposition 4. We have

$$D(\pi_D||\tilde{\pi}_T^{(1)}) \leq I_S(A; Y_D|Y_S).$$

In the discrete case, additionally, the r.h.s. can be bounded by

$$I_S(A; Y_D|Y_S) \leq H(Y_D|Y_S).$$

The above result implies that the proposed proxy and the demonstrator's policy are the same, when there exist deterministic relationship between the observation sets. Next result goes beyond the policies and looks at the overall behavior of the system induced by this policy.

Proposition 5. The proposed proxy in (9) implies that the KL-divergence in (7) is bounded by $D(\tilde{\pi}_T^{(1)}||\pi_D)$.

Second case: In this case, the spectator and the demonstrator have the same set of sensors in the source domain, i.e., $P_S(Y_S|X) = P_S(Y_D|X)$ but the imitator can have different sensors in the target domain. Optimizing an upper bound of (7) that is described in the Supplement gives the following policy to the imitator,

$$\tilde{\pi}_T^{(2)}(a|y_T) \propto \exp\left(\sum\!\!\!\!\!\!\int_{y_S} p(y_S|y_T) \log p_S(a|y_S)\right).$$

Proposition 6. The proposed policy in this case will lead to the following upper bound for (7),

$$\sum_{a,y_T,y_S} p(y_S|y_T)p_T(y_T)\tilde{\pi}_T^{(2)}(a|y_T) \log \frac{\tilde{\pi}_T^{(2)}(a|y_T)}{p_S(a|y_S)}.$$

Note that in an extreme setting when Y_S is determined uniquely from Y_T, it is straightforward to show that the upper bound in Proposition 6 becomes zero. Thus, the proposed proxy leads to the demonstrator's behavior.

Third case: This is the general case where all sensors can be different. Note that Example 3 belongs to this case. Here, we propose the following policy for the imitator

$$\tilde{\pi}_T^{(3)}(a|y_T) \propto \exp\left(\sum\!\!\!\!\!\!\int_{x} p(x|y_T) \log \tilde{p}(a|x)\right),$$

where

$$\tilde{p}(a|x) := \sum\!\!\!\!\!\!\int_{y} p_S(a|Y_S = y)p_S(Y_S = y|x).$$

We introduced the other two cases since they occur frequently in different applications and we can derive theoretical bounds for them.

8 Experiments

In this section we perform experiments for some of the methods proposed in Sections 5, 6 and 7.

8.1 Action-effect learning task

Setup: In this experiment, we test two of our methods for the action-effect transfer learning task: Algorithm 2 and the proxy in (6) (more specifically: a sample-level version of it for the linear case). We use the real-world data set "highD" (**?**) that consists of recordings by drones that flew over several highway sections in Germany (mentioned in Example 1). From this data set, we selected all situations, where there is a lead car – the demonstrator (this is a different setup than Example 1[10]) – and a following car on the same lane (which are less than 50m from each other, and have speed at least 80km/h). Here X is distance, velocities, and acceleration of the follower; A is the acceleration of the demonstrator; and Z is the acceleration of the follower, 1.5 seconds later.

Furthermore, the source domain's Y_S is generated by a randomly drawn matrix F applied to X plus Gaussian noise (as in (4)). This semi-real approach allows us to have ground truth samples from $P(Z, A, X) = P_T(Z, A, Y_T)$, i.e., the target domain (recall our Assumption 1). We apply the two methods on training samples from the source domain $P_S(Z, A, Y_S)$ up to length 20000, and calculate the means (over 20 different data and synthetic noise samples) squared error on separate test samples of length 1000 from $P(Z, A, X)$.

[10]While this is the data set mentioned in Example 1, here we do not consider the indicator lights, since for them we would not have the ground truth.

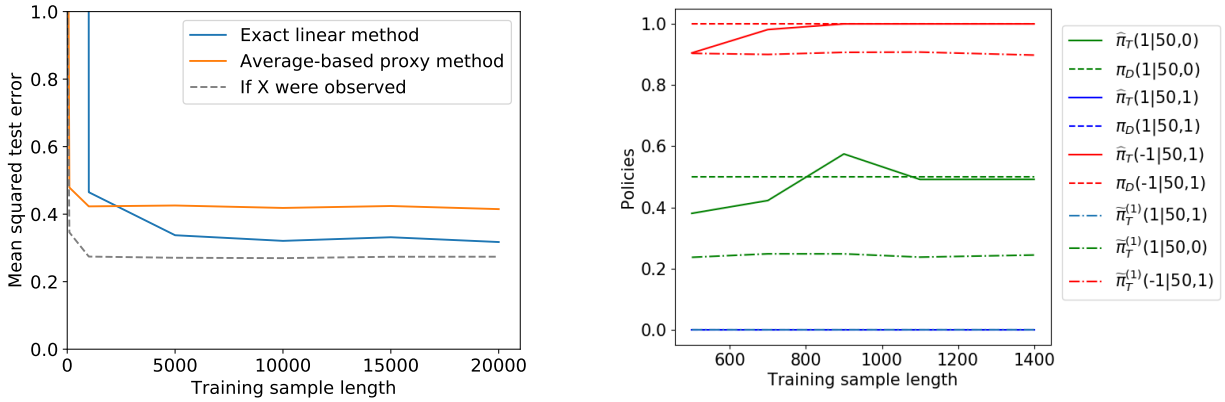

Figure 3: **Left:** Outcome for the action-effect learning experiment. Our exact linear transfer method (Algorithm 2) has higher *variance*, but outperforms the average-based proxy method (sample-level version of (6) for linear case), which can be seen as a baseline, for *long enough samples*. We also plot what could be achieved if X was fully observed in the source domain, as a lower bound. **Right:** Learned policies for the imitation learning experiment: the true policy π_D, the policy from the method in Section 7.1, $\hat{\pi}_T$, and the corresponding proxy $\tilde{\pi}_T^{(1)}$. The three policies are evaluated at three different points $(a|V_o, b_o) \in \{(1|50,0), (1|50,1), (-1|50,1)\}$.

Outcome: The outcome for this experiment is *depicted and discussed* in Figure 3.

8.2 Imitation learning task

Setup: In this experiment we simulated the driving scene illustrated in Figure 1. The observation set of the demonstrator Y_D contains the speed $v_o \in \{40, 45, ..., 60\}$ km/h and the indicator light $b_o \in \{0, 1\}$ of the lead vehicle. The imitator only gets to see a noisy observation of the demonstrator's speed, i.e., $Y_S = v_d + N$, where $N \sim \mathcal{N}(0, 1/4)$. Actions are $-1, +1, 0$ denoting speed reduction by 5km/h, increasing it by 5km/h, and keep the same speed, respectively. In this experiment, we assumed $Y_D = Y_T$.

We defined the demonstrator's policy to reduce the speed when the indicator of the other vehicle is on $b_o = 1$ and increase its speed or keep the same speed when $b_o = 0$. Note that the classical imitation learning approach will fail in this setting since $Y_T \neq Y_S$.

We applied Algorithm 1 plus a criterion to obtain the policy $\tilde{\pi}_T^{(1)}$ for the imitator This criterion (that is described in the supplement) ensures that the imitator neither increases its speed when $b_o = 1$ nor decreases its speed with the same probability when $b_o = 0$. We formulated this as a linear programming.

Outcome: Figure 3 compares the true policy π_D, the policy from the method in Section 7.1, $\hat{\pi}_T$, and the corresponding proxy $\tilde{\pi}_T^{(1)}$ for different sample sizes.

9 Conclusions

Sensor-shift is a significant problem in learning from demonstrations. In this work, we proposed a principled and general framework to address it, based on causal modeling. We developed novel algorithms that uniquely identify or constrain/approximate the relevant causal effects, and established theoretical guarantees. The take away message is that the relevant causal relationships may still be identifiable, even if the demonstrator, spectator and target agent have different sensors.

Repellendus illum consectetur totam numquam corporis facere ab sequi aliquid, ea dolorem nobis voluptatibus suscipit, incidunt aut labore cumque earum vitae amet reiciendis assumenda et nesciunt, beatae ex velit adipisci placeat quisquam repellat cupiditate totam voluptatem, expedita odio repudiandae?Quisquam earum provident aliquid laboriosam velit error atque ducimus ut cupiditate, similique repellat repellendus tempora beatae error aut blanditiis velit asperiores harum ipsum, nesciunt rem odio incidunt molestiae omnis ullam autem.Earum eligendi illum laboriosam, sint fuga modi enim beatae facilis, cumque fugit recusandae.Accusamus eum maxime placeat labore, praesentium impedit earum obcaecati, perferendis voluptate dolore qui culpa, molestiae voluptatibus sunt nesciunt provident quod nemo magnam fuga totam quasi aspernatur.Quisquam modi veritatis, tempora repudiandae voluptatum.Aliquam laboriosam facilis quas eius odio nam similique, odit dicta illum adipisci ratione perspiciatis provident delectus eveniet, porro facere cumque magnam iusto quaerat vitae deleniti saepe vero veniam?Pariatur delectus doloremque qui optio ex at provident vero dolorem in, necessitatibus odio nemo magni ea reiciendis accusamus laudantium consectetur voluptate eos debitis, necessitatibus fugit distinctio provident nesciunt doloribus odio quam architecto reiciendis esse voluptatum.Vitae debitis perspiciatis commodi, ipsum sapiente repellat consequuntur deserunt ratione corporis temporibus quidem veniam quasi repudiandae?Maxime possimus praesentium doloribus, sit laborum voluptatem labore repellendus qui libero provident, saepe qui exercitationem iure dolores nesciunt optio

quas alias nam perferendis, doloribus beatae aperiam modi aliquid eligendi, omnis nisi beatae.Sapiente libero vel eaque architecto magnam provident