IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Generating and Transferring Priors for Causal Bayesian Network Parameter Estimation in Robotic Tasks*

Maximilian Diehl and Karinne Ramirez-Amaro

***Abstract*—Robots acting in human environments will often face new situations and can benefit from transferring prior experience. Priors could enable robots to handle new tasks zero-shot and help prevent failures, which can be particularly costly in real robot applications. Due to their interpretable nature, causal Bayesian Networks (CBN) are popular for modeling cause-effect relations between semantically meaningful environment features and their effects on action success. While the CBN structure is often intuitively transferable to a new context, its probability distribution might change, requiring data-intensive relearning. In this work, we propose three strategies that utilize semantic similarity and relatedness between the variables of two CBNs to generate and transfer informed CBN distribution priors. We evaluate the parameter prior accuracy in five different transfer scenarios, including sim-2-real, transferring parameters to more complex tasks with a larger number of parameters and even between two different tasks, which is particularly challenging. We show that the priors lead to better distribution estimates, particularly under a limited amount of new experiments, and improve the robot's ability to predict and prevent action failures by up to 50%.**

***Index Terms*—Learning from Experience, Probability and Statistical Methods, Transfer Learning**

I. Introduction

ROBOTS frequently encounter novel situations in human environments, necessitating task execution in varied environments and with diverse objects. Relearning the tasks would typically require a substantial amount of data, which poses a challenge because real robot task executions are time-intensive. Furthermore, execution failures could lead to potential damage to the environment or robotic platforms. To address these issues, the robotic community proposed to enhance the robot's adaptability by leveraging prior knowledge for zero-shot handling of new tasks and increasing sample efficiency for adapting to novel situations through knowledge representations like ontologies [1], [2] or data-driven methods like neural networks [3], [4]. However, ontological models are often limited in their adaptability in dynamic environments, and neural networks rely primarily on sensory features, making failure analysis more complex due to a lack of semantically meaningful features. As an alternative, causal Bayesian Networks (CBNs) have recently gained increased attention [5], [6].

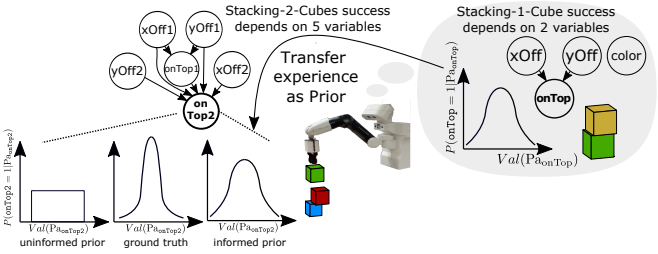

Fig. 1. Shows an example of a CBN parameter transfer from a known task (right) to a more complex target task (left). The parameters, represented by the different distributions, can be interpreted as the stacking success probability, given different combinations of x/y-Offsets between the cubes.

CBNs can model cause-effect relationships between semantically meaningful environment features and their effects on task execution success. As a result, they have been used for explaining failures [7] and predicting and preventing failures [8]. However, CBN learning methods are data-consuming due to their statistical nature [9]. In this paper, we, therefore, propose to utilize and transfer CBN models of known tasks to related but novel tasks (e.g., transfer the task of stacking one cube to two cubes). Learning a CBN consists of: 1) Discovering variable relationships (structure learning), and 2) determining the parameters defining conditional probability distributions for each CBN variable (parameter learning). Particularly relevant are the conditional probability distributions of variables like `onTop1` representing the stacking success of the first cube (see Fig. 1). They enable us to predict task success given various parent variable combinations (e.g., in Fig. 1 centered stacking results in a high success chance, while large x/y-Offsets lead to lower success chances).

The CBN structure can often be transferred to related tasks, as shown in Fig. 1, where both stacking tasks rely on x/y-Offset. However, stacking two cubes is more challenging, as its success also depends on the alignment of the first stacking action. This complexity is reflected in the conditional probability distribution used to predict task success. With an increase in parent variables, the number of all possible combinations between the parent variable values increases, thus making the transfer of the parameters more complicated. Hence, the primary question we address in this paper is how to effectively generate and transfer distribution parameters from

Manuscript received: 27 July 2023; Revised 19 October 2023; Accepted 13 November 2023.

This paper was recommended for publication by Editor Tetsuya Ogata upon evaluation of the Associate Editor and Reviewers' comments.

*This work is supported by Chalmers AI Research Centre (CHAIR).

Maximilian Diehl and Karinne Ramirez-Amaro. Faculty of Electrical Engineering, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden. {diehlm, karinne}@chalmers.se

Digital Object Identifier (DOI): see top of this page.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

one CBN to a related one. We assume that the structure is already known (e.g., for the discussed tasks, the structure has been determined in prior work [7], [8]). Our goal is to obtain accurate parameter estimates for the target CBNs, even when we have limited or no data instances of the target task available.

Our main contributions are three parameter transfer strategies that make use of the semantic[1] variable description of the CBNs to generate informed distribution priors and transfer them to new CBNs based on 1) semantically identical CBN features (Direct-Replacement), 2) semantically related features (Cross-Replacement), or 3) a combination of semantically related and similar features (Minimal-Replacement). Furthermore, we evaluate the effectiveness of our proposed transfer strategies for a range of increasingly difficult CBN transfer scenarios, where i) the CBN structure remains but the distribution changes, ii) the CBN structure and distribution change but the task remains the same (in simulation and reality), iii) the CBN structure remains the same but the manipulated objects change and iv) the CBN structure, distribution and the task change. We are particularly interested in showing the impact of the priors on the decision-making abilities of the robot in novel situations, with a limited amount of new experiments on the target task. Therefore, we use the priors as input for a method [8] that relies on accurate probability distribution estimates for predicting and preventing task execution failures[2].

II. Related Work

Causal Bayesian Networks are gaining increasing attention in robotics [5]. Recently, CBNs were used to generate contrastive explanations of robot task execution failures [7]. Brawer et al. present a causal approach to tool affordance learning [6]. Other works use BNs to learn statistical dependencies between object attributes and grasp actions in simulation [10]. These works either assume that the observed connections are not too data-intensive or have enough samples. None of these works discuss the data requirements or look into the possibility of transferring prior experience in the form of data priors to new robot tasks.

Some works transfer robot task executions to novel situations based on ontologies. Bayer et al. learn probabilistic action effects of dropping objects into different containers [1] and generalize the success probability predictions for various objects, like bowls and bread boxes. Another approach [2] combines an ontology with edge weights that indicate the generalization strength between object classes for grasping objects and stowing them in a drawer. While we share the objective of learning generalizable prediction models, we particularly learn which object properties and environment preconditions are relevant for task transfer to generalize from one task to another more reliably. In addition to ontologies, data-driven models, such as neural networks [3], [4], have proven successful in various transfer tasks. In [3], the authors learn the reward function from related tasks in an inverse reinforcement learning problem. Another work [4] demonstrates domain adaptation alongside a data-driven, vision-based grasping approach to transfer grasps from simulated to novel real-world objects. Although these approaches showcase the effectiveness of data-driven models, their applicability in human-centered scenarios is problematic as generalization failures become challenging to analyze due to the lack of semantically meaningful network features, which we address by using CBNs.

Priors have been proposed as a solution to alleviate the substantial data requirements in (causal) Bayesian network (BN) learning. Various approaches involve transferring the BN variable structure, as seen in prior work [11], [12], supporting our assumption that the BN structure can be intuitively transferred to new tasks. In our work, we specifically focus on transferring BN probability distribution parameters. Our approach is closely related to methods that enable the transfer of BN parameters from multiple prior BNs to a new target network [13], [14]. These methods use a measure of relatedness to identify which prior BN variables are closely related to target BN variables, then transfer a weighted combination of their conditional probability distribution parameters. Notably, these approaches assume that the prior and target variables have the same number of probability distribution parameters. This assumption does often not hold. For instance, the 2-Stack task has more parameters than the 1-Stack task (see Fig. 1). Furthermore, particularly [14] does not assume variable correspondences between two tasks. That means variable names (or the variables' meanings) are not used to aid the parameter transfer. However, we particularly want to take advantage of variables that are shared between tasks (such as the x-offset, which is part of both the 1-Stack BN model and the 2-Stack BN model).

Some related works focus on reducing data requirements by selectively sampling sensitive network parameters [9]. However, they do not explore data priors for learning new tasks or zero-shot applications. Data extension methods [15] attempt to alleviate the need for extensive datasets in BN parameter learning by incorporating non-optimal data, using constraints like proximity to target task parameters to enhance parameter estimation in new tasks. While this approach aligns with our BN parameter transfer problem in theory (by viewing 1-Stack task data as an extension for the 2-Stack task), it lacks a method for establishing variable correspondences in cases where the extended data contains different or related variables. This is a challenge our proposed parameter transfer approaches address.

III. Our Approach to CBN Parameter Transfer

In this section, we discuss our proposed parameter transfer strategies and how they are placed within the traditional CBN learning pipeline. We first introduce traditional structure learning (Sec. III-A1) and the two main parameter estimation methods (Sec. III-A2): Maximum Likelihood Estimation (learning from scratch) and Bayesian Estimation (learning with a prior). Both parameter estimation methods suffer from

[1]Semantics refers to the underlying meaning of each CBN variable.

[2]Please find a video of our experiments under the following link https://youtu.be/F4zCNoohzOs

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

an exploding data demand [16] with increasing network complexity. For instance, doubling the number of dependent variables (e.g., transfer from `onTop1` to `onTop2` in Fig. 1) leads to a quadratic increase variable value combinations, and thus to a quadratic increase in data samples for the same level of estimation confidence. In Sec. III-B, we, therefore, show how the Bayesian Estimation can be reformulated to incorporate parameter priors from simpler CBNs, which leads to the formulation of our three proposed parameter transfer strategies.

A. Preliminaries

Formally, CBNs are defined as *directed acyclic graphs* (DAG) $\mathcal{G} = (\boldsymbol{X}, A)$, where the nodes $\boldsymbol{X} = \{X_1, X_2, ..., X_N\}$ are a set of N random variables $X_i \subseteq \boldsymbol{X}$ and A is the set of arcs [17] that describe the causal connections between the variables. We model a robot task T, such as dropping a sphere into a container or stacking cubes as CBN, where $\boldsymbol{X}$ represent the task variables. Based on the dependency structure of the DAG and the *Markov property*, the *joint probability distribution* of a causal BN can be factorized into a set of *local probability distributions*, where each random variable X_i only depends on its direct parents Pa_{X_i}:

$$P(X_1, X_2, ..., X_N) = \prod_{i=1}^{n} P(X_i|\text{Pa}_{X_i}) \tag{1}$$

Traditionally, BNs are learned in two steps: *Structure Learning* and *Parameter Estimation*.

1) Structure Learning: To learn causal relations in $\mathcal{G} = (\boldsymbol{X}, A)$, we employ the PC[3] algorithm [18], a constraint-based method that utilizes statistical tests to identify conditional independence relations from the data [19]. While PC is one of several eligible structure learning algorithms for this purpose [20], it's worth noting that many of these algorithms require discrete variables as parents for other discrete variables [21]. To address this, we perform quantile discretization on all continuous variables in $\boldsymbol{X}$, dividing each X_i into intervals with equal data sample counts.

2) Parameter Estimation: Once $\mathcal{G}$ is obtained, the local probability distributions $P(X_i|\text{Pa}_{X_i})$ (Eq. 1) can be represented using parameters $\boldsymbol{\theta}$, with their interpretation depending on the probability distribution type of X_i. In our case, we model each X_i as a multinomial random variable due to variable discretization. Thus, $\boldsymbol{\theta}$ takes the form of a table. In this table, each $\theta_{x|\boldsymbol{u}} \in \boldsymbol{\theta}$ represents the probability that $\theta_{x|\boldsymbol{u}} = P(X_i = x|\text{Pa}_{X_i} = \boldsymbol{u})$ for every $x \in Val(X_i)$ and $\boldsymbol{u} \in Val(\text{Pa}_{X_i})$, where $Val(X_i)$ signifies all possible values of X_i, and $Val(\text{Pa}_{X_i})$ encompasses all potential value combinations of Pa_{X_i}. The cardinality of $|\boldsymbol{\theta}| = |x||\boldsymbol{u}|$ grows with the number of discretization intervals for each $\boldsymbol{X}$.

There are two main parameter estimation approaches: *Maximum Likelihood estimation* (MLE), and *Bayesian estimation* [22]. For both, we assume that we have obtained a training data set $\mathcal{D}$, which consists of K independent and identically distributed (iid) samples $\boldsymbol{\xi}$. Each sample ξ is a fully observed instance of all network variables $\boldsymbol{X}$.

[3]PC is named after its authors Peter Spirtes and Clark Glymour.

Maximum Likelihood Estimation aims to find the optimal parameters $\hat{\boldsymbol{\theta}} \in \Theta$ of the parametric model $P(\mathcal{D} : \boldsymbol{\theta})$ that represent the unknown CBN distribution $P(X_1, X_2, ..., X_N)$. Due to the global likelihood decomposition [16], the optimal CBN parameters can be obtained for each $\hat{\theta}_{x|\boldsymbol{u}}$ individually by maximizing the likelihood function

$$L_{X_i}(\theta_{X_i|\text{Pa}_{X_i}} : \mathcal{D}) = \prod_{\boldsymbol{u} \in Val(\text{Pa}_{X_i})} \left[\prod_{x \in Val(X_i)} \theta_{x|\boldsymbol{u}}^{M[\boldsymbol{u},x]} \right], \tag{2}$$

where $M[\boldsymbol{u}, x]$ is the number of times $X_i[k] = x$ and its parent variables $\boldsymbol{U}[k] = \boldsymbol{u}$ for $k \in K$ in $\boldsymbol{\xi}$. Then, L_{X_i} is maximized by maximizing each of the $\theta_{x|\boldsymbol{u}}^{M[\boldsymbol{u},x]}$, individually:

$$\hat{\theta}_{x|\boldsymbol{u}} = \frac{M[\boldsymbol{u}, x]}{M[\boldsymbol{u}]}. \tag{3}$$

Bayesian Parameter Estimation treats each θ as a random variable and, unlike MLE, encodes prior information about θ, leading to the formulation of the posterior distribution

$$P(\theta|\mathcal{D}) = \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})}. \tag{4}$$

With the assumption that parameter priors for each of the local conditional probability distributions of Eq. 1 are independent, we can show that the posterior distribution of the CBN parameters $\boldsymbol{\theta}$ can be decomposed into

$$P(\boldsymbol{\theta}|\mathcal{D}) = \prod_{i} \prod_{\boldsymbol{u} \in Val(\text{Pa}_{X_i})} P(\boldsymbol{\theta}_{X_i|\boldsymbol{u}}|\mathcal{D}). \tag{5}$$

Furthermore, in the case of a single multinomial variable X_i with J possible outcomes, we can derive a predictive model for observing a new data sample $\xi[K+1]$ where $X_i[K+1] = x_j$ with $j \in J$, given that the parent variables $\boldsymbol{U}[K+1] = \boldsymbol{u}$ and the previously observed data $\mathcal{D} = \{\xi_1, \xi_2, ..., \xi_K\}$, as

$$P(X[K+1] = x_j|\boldsymbol{U}[K+1] = \boldsymbol{u}, \mathcal{D}) = \frac{\alpha_{x_j|\boldsymbol{u}} + M[x_j, \boldsymbol{u}]}{\alpha_{\boldsymbol{u}} + M[\boldsymbol{u}]}, \tag{6}$$

by modelling the prior distribution $P(\boldsymbol{\theta}_{X_i})$ as a Dirichlet distribution with hyperparameters $\alpha_{x_1|\boldsymbol{u}}, \alpha_{x_2|\boldsymbol{u}}, ..., \alpha_{x_J|\boldsymbol{u}}$. The advantage of using the Dirichlet distribution is that the hyperparameters can be interpreted as an imaginary count [16] of successful task executions in a prior data set $\mathcal{D}'$. More formally, we can set $\alpha_{x_j|\boldsymbol{u}} = \alpha[x_j, \boldsymbol{u}]$ where $\alpha[x_j, \boldsymbol{u}]$ is the number of times $X_i = x_j$ and $Val(\text{Pa}_{X_i}) = \boldsymbol{u}$ in $\mathcal{D}'$. Analogously, $\alpha_{\boldsymbol{u}} = \alpha[\boldsymbol{u}]$ where $\alpha[\boldsymbol{u}]$ is the number of times that $Val(\text{Pa}_{X_i}) = \boldsymbol{u}$ in $\mathcal{D}'$.

B. Our approach for generating causal BN parameter priors

Our approach addresses the challenge of deriving an imaginary prior count $\frac{\alpha[x_j,\boldsymbol{u}]}{\alpha[\boldsymbol{u}]}$ in Eq. 6 from a CBN model of a prior task, which is already known or can be obtained with significantly less effort (fewer samples and less time) due to a smaller cardinality $|\boldsymbol{\theta}|$. To formalize the generation of parameter priors, we define a prior task T' as a causal BN structure $\mathcal{G}'$, set of network variables $\boldsymbol{X}'$, set of parameters $\boldsymbol{\theta}'$ and a dataset $\mathcal{D}'$. Furthermore, we define the target task

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

as T, with its own associated causal BN structure $\mathcal{G}$, network variables $\boldsymbol{X}$, parameters $\boldsymbol{\theta}$ and dataset $\mathcal{D}$, where generally $\mathcal{G}' \neq \mathcal{G}$ and $\mathcal{D}' \neq \mathcal{D}$ but $\boldsymbol{X}' \cap \boldsymbol{X} \neq \emptyset$. We denote $X' \in \boldsymbol{X}'$ as the variable whose parameters $\boldsymbol{\theta}'_{\boldsymbol{X}'}$ we want to transfer to $X \in \boldsymbol{X}$. In this paper, we then propose to use T' as prior information for learning the BN parameters of T.

Challenge 1: large number of pseudo counts A naive parameter transfer based on the notion of pseudo counts in Eq. 6 is problematic because $\alpha[\boldsymbol{u}]$ might be much larger than $M[\boldsymbol{u}]$, due to the easier availability of prior data. Thus samples $\xi \in \mathcal{D}$ would barely play a role. We therefore reformulate Eq. 6 as

$$\frac{\alpha_{x_j|\boldsymbol{u}} + M[x_j, \boldsymbol{u}]}{\alpha_{\boldsymbol{u}} + M[\boldsymbol{u}]} = \frac{w_{\text{prior}}\theta'_{x_j|\boldsymbol{u}} + M[x_j, \boldsymbol{u}]}{w_{\text{prior}} + M[\boldsymbol{u}]}, \tag{7}$$

where $\theta'_{x_j|\boldsymbol{u}} = \frac{M[x_j,\boldsymbol{u}]}{M[\boldsymbol{u}]}$ is based on the prior data $\mathcal{D}'$. Eq. 7 allows us to deliberately control the importance of the prior. For example, by choosing $w_{\text{prior}} = M[\boldsymbol{u}]$, we would weigh the prior and the new data equally.

Challenge 2: Difference in number of parameters So far in Eq. 7 we assume that we can find a corresponding prior parameter $\theta'_{x_j|\boldsymbol{u}}$ for each $M[x_j, \boldsymbol{u}]$. However, if either the number of parent variables is not consistent $|\text{Pa}_{X'}| \neq |\text{Pa}_X|$ (where $|.|$ denotes the cardinality) or the number of values is different $|Val(\text{Pa}_X)| \neq |Val(\text{Pa}_{X'})|$ we could end up in a situation where $|\boldsymbol{u}'| \neq |\boldsymbol{u}|$, which means we won't find a $\theta'_{x_j|\boldsymbol{u}}$ for every $M[x_j, \boldsymbol{u}]$. We therefore propose a generalization of Eq. 7 as

$$\frac{\alpha_{x_j|\boldsymbol{u}} + M[x_j, \boldsymbol{u}]}{\alpha_{\boldsymbol{u}} + M[\boldsymbol{u}]} = \frac{w_{\text{prior}} f(\theta'_{x_j|\boldsymbol{u}'}) + M[x_j, \boldsymbol{u}]}{w_{\text{prior}} + M[\boldsymbol{u}]}, \tag{8}$$

where $f(\theta'_{x_j|\boldsymbol{u}'}) = \frac{M[x_j,\boldsymbol{u}']}{M[\boldsymbol{u}']}$ is the ML estimate of $\theta'_{x_j|\boldsymbol{u}'}$ for $\mathcal{D}'$. $\boldsymbol{u}'$ is defined based on one of the following three proposed prior generation and transfer approaches:

1) Direct-Replacement: $\boldsymbol{u}' = \boldsymbol{u}_{\text{Pa}_{X'} \cap \text{Pa}_X} \in Val(\text{Pa}_{X'} \cap \text{Pa}_X)$. That means the discretization intervals of the non-empty variable intersection set $\text{Pa}_{X'} \cap \text{Pa}_X \neq \emptyset$ in $\boldsymbol{u}'$ are set to the discretization intervals of the similar subset in $\boldsymbol{u}$. See Ex. 1 for an example.

2) Cross-Replacement: $\boldsymbol{u}' = \boldsymbol{u}_{rel(\text{Pa}_{X'}, \text{Pa}_X)} \in Val(rel(\text{Pa}_{X'}, \text{Pa}_X))$, where $rel()$ returns a subset of variables $X'_{\text{rel}} \subset \text{Pa}_{X'}$ that is related to a subset of variables $X_{\text{rel}} \subset \text{Pa}_X$. We denote variables as related if they are semantically similar but not the same, e.g., `xOff1` and `xOff2` both describe the x-Offset but between different cubes. We currently provide the related variables manually but want to investigate how to automate this step in future work, for example, based on partly matching variable names and the similarity of the variables' distribution and value range. See Ex. 1 for an example.

3) Minimal-Replacement:

$$\boldsymbol{u}' = \begin{cases} \boldsymbol{u}'_{\text{direct}} & \text{if } \theta'_{x_j|\boldsymbol{u}'_{\text{direct}}} < \theta'_{x_j|\boldsymbol{u}'_{\text{cross}}} \\ \boldsymbol{u}'_{\text{cross}} & \text{else} \end{cases}, \tag{9}$$

where $\boldsymbol{u}'_{\text{direct}} = \boldsymbol{u}_{\text{Pa}_{X'} \cap \text{Pa}_X}$ and $\boldsymbol{u}'_{\text{cross}} = \boldsymbol{u}_{rel(\text{Pa}_{X'}, \text{Pa}_X)}$. This strategy regards both Direct- and Cross-Replacement parameter candidates and uses the smaller one. See Ex. 1 for an example.

Example 1: Assume the robot in Fig. 1, aims to predict $P(\texttt{onTop2} = 1|\texttt{xOff1} = 0.5\text{cm}, \texttt{yOff1} = 1\text{cm}, \texttt{xOff2} = 1.5\text{cm}, \texttt{yOff2} = 2\text{cm}) = P_{\text{Example1}}$ based on knowing the distribution $P(\texttt{onTop}|\texttt{xOff}, \texttt{yOff})$. **Direct-Replacement** would determine the parameter prior as $P_{\text{Example1}} = P(\texttt{onTop} = 1|\texttt{xOff} = 0.5\text{cm}, \texttt{yOff} = 1\text{cm})$, thus only considering the offset between the first cube and the base cube. **Cross-Replacement** would disregard the offset of the first cube and determine the parameter prior based on the offset of the second cube only $P_{\text{Example1}} = P(\texttt{onTop} = 1|\texttt{xOff} = 1.5\text{cm}, \texttt{yOff} = 2\text{cm})$. **Minimal-Replacement** calculates the prior probabilities based on both above-mentioned strategies, and then uses the one with the smaller probability value.

IV. Experiment Description

In our experiments, we aim to assess how well the obtained parameters priors approximate the ground truth parameters of the target task. We are interested in assessing which prior transfer strategy works best and how to choose the weight factor w_{prior}. Furthermore, we want to compare if parameter estimation with priors (Bayesian Estimation) is more efficient than learning the parameters from scratch (MLE).

A. Experiment Environments

In our experiments, we use the tasks of stacking two cubes in simulation and reality (2-Stack) and dropping a sphere into different containers in simulation (SphereDropping).

1) 2-Stack Environment: Consists of *CubeUp1*, *CubeUp2* and *CubeDown* (see Fig. 2-a). The goal is to place *CubeUp2 on top* of *CubeUp1* and *CubeUp1 on top* of *CubeDown*. All cubes have a side length of 5cm.

For the 2-Stack-Simulation task, we conduct 1,000,000 2-Stack experiments in a Unity3d simulation environment without a simulated robot. Refer to Tab. I-a for a list of all task variables $\boldsymbol{X}_{\text{Stack}}$ and their respective variable distribution and ranges. For each stacking experiment, the cubes are initialized with respect to sampled variable values. After releasing the cubes, $\texttt{onTop}_i$ for $i = \{1, 2\}$, is automatically determined.

For the 2-Stack-Reality task, we conduct a total of 360 2-Stack experiments utilizing the TIAGo robot (see Fig. 2-b), with 40 samples per parameter. Environment, goal, and variables remain similar to the 2-Stack-Simulation experiments, however, to obtain enough ground truth data, we reduce the number of possible variable values to $\texttt{xOff}_i = \{0, 0.9, 2.2\}$ (in cm) (which are the middle points of the obtained variable intervals reported in Tab. I-a), $\texttt{yOff}_i = 0$ and $\texttt{dropOff}_i = [0.5, 10]$. To achieve consistent stacking positions, we predetermine arm joint positions that lead to each possible variable combination, manually feed the cubes into the gripper, and determine the stacking success after the robot opens its gripper.

In this paper, we consider the causal structure for both tasks (shown in Fig. 3 a) and b)) as they are obtained from our previous work [7], [8]. We specifically focus on the challenging problem of transferring the causal BN parameters.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

Variable	Meaning
$\texttt{xOff}_2$	$\texttt{cubeUp}_2.x - \texttt{cubeUp}_1.x$ (distance between cube centers)
$\texttt{yOff}_2$	$\texttt{cubeUp}_2.y - \texttt{cubeUp}_1.y$
$\texttt{dropOff}_2$	$(\texttt{cubeUp}_2.z - \texttt{cubeUp}_1.z)$ $-\texttt{cubeLength}$ (distance between cube surfaces)
$\texttt{onTop}_2$	indicates if $\texttt{cubeUp}_2$ is on top of $\texttt{cubeUp}_1$ after the stacking process
$\texttt{cubeColor}_2$	denotes cube color

Variable	Meaning
xOff	$\texttt{sphere}.x - \texttt{container}.x$ (distance between sphere and container center)
yOff	$\texttt{sphere}.y - \texttt{container}.y$
ContainerDiameter	denotes the diameter of the container
ContainerHeight	denotes the height of the container
ContainerCurvature	denotes if the container has a curvature
ContainerColor	denotes the color of the container
inside	denotes if the sphere ended up in the container

a) 2-Stack-Simulation **b) 2-Stack-Reality** **c) SphereDropping-Simulation**

Fig. 2. a) defines the causal BN variables for the 2-Stack task (variables that describe the stacking relationship between *CubeUp1* and *CubeDown* are defined analogously), b) shows a snapshot of the TIAGo robot conducting one iteration of the 2-Stack-Reality experiment and c) defines the causal BN variables for the SphereDropping task.

2) SphereDropping Environment: Contains a *Sphere* and one of three possible *Containers*: plate, bowl, or glass (see Fig. 2-c). The goal is to drop the sphere into the containers. Refer to Tab. I-b for a list of all task variables $\boldsymbol{X}_{\text{Drop}}$. We conduct 1,000,000 SphereDropping experiments in a Unity3d simulation environment (around 333,333 samples per container) without a simulated robot. For each dropping experiment, the sphere and container are initialized with respect to sampled variable values. After releasing the sphere, `inside` is automatically determined. We do not directly set `containerHeight/Size` but manipulate these variables via a $\texttt{scalingFactor} \sim \mathcal{U}_{[0.4,1]}$. In Unity3d, the scaling factor can be utilized to manipulate the size of objects. We use the same scaling factor in all three dimensions; thus, the height (`containerHeight`) to diameter (`containerDiameter`) ratio is constant for each object type, respectively. The sphere has a fixed diameter of 6.6cm, and we choose a constant dropping height of 0.4m.

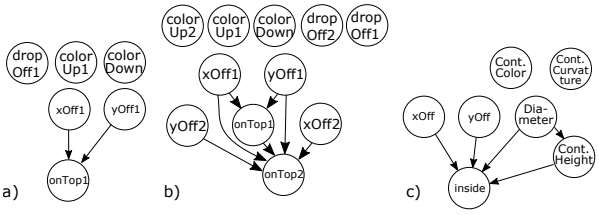

Fig. 3. Visualizes the causal relations for a) 1-Stack task, b) 2-Stack task and c) the SphereDropping task.

B. Prior Transfer Scenarios

We define five different prior transfer scenarios as detailed in Tab. II and conduct the following steps 20 times for every scenario:

1) We randomly split the available data into three datasets: $D', D, D_{\text{GroundTruth}}$, where $D \cap D_{\text{GroundTruth}} = \emptyset$.
2) We determine prior task success probability $\theta'_{x_g=1|u'}$, where x_g is the task success variable (e.g., x_g is $\texttt{onTop}_1$ in case of the 1-Stack task).
3) We apply one or several of the proposed transfer methods to obtain $f(\theta'_{x_j|\boldsymbol{u}'})$.
4) We split D into 20 subsets and apply MLE and the Bayesian Estimation on an increasing number of subsets. Note that for the Bayesian Estimation, we don't calculate a new prior after increasing the size of D with a new subset. Our prior is always only based on the prior task T'.

C. Experiment Evaluation Metrics

We introduce three metrics to measure the outcome of our experiments:

1) Max-Difference: $\max_{x \in Val(X_i), \boldsymbol{u} \in Pa_{X_i}} (|\boldsymbol{\theta}_{x|\boldsymbol{u}} - \boldsymbol{\theta}^*_{x|\boldsymbol{u}}|)$, with $X_i \in \boldsymbol{X}$ and $\boldsymbol{\theta}^*$ denoting the ground thruth.

2) Mean-Difference: $\sum_{x \in Val(X_i)} \sum_{\boldsymbol{u} \in Pa_{X_i}} \frac{|\boldsymbol{\theta}_{x|\boldsymbol{u}} - \boldsymbol{\theta}^*_{x|\boldsymbol{u}}|}{|\boldsymbol{u}|}$.

3) Failure Prevention: Represents the ground truth task success probabilities determined after applying the failure prevention method [8] based on the estimated parameters as input. This method employs the obtained CBN structure and estimated parameters to predict the likelihood of action failure and its underlying cause based on the current CBN variable assignment. If the predicted success probability for a task falls below the threshold of $\epsilon = 0.8$, the method initiates a search for the nearest variable values with a predicted success chance greater than ϵ. For instance, if the method predicts that stacking a cube will be placed too far to the right, our algorithm suggests the closest x-Offset value to ensure successful task execution.

Ideally, parameter estimates from MLE or Bayesian Estimation should closely match ground truth parameters, assessed using Max- and Mean-Difference metrics. While Mean-Difference offers a general picture of parameter accuracy, Max-Difference is crucial for understanding the maximum potential error. In this paper, our focus is on scenarios where the robot has limited target task data (e.g., fewer new target task executions than parameters) or potentially has to transfer prior knowledge zero-shot. In such cases, Max- and Mean-Difference metrics may not indicate how useful these estimations are for the robot's decision-making. Setting all parameters to 0.5 would minimize the Max- and Mean-Difference when $|D| = 0$, but it does not help the robot derive a conclusion on whether an action is going to succeed or not. Therefore, we introduce the Failure Prevention metric. An informed prior might perform worse in Max and Mean metrics but can be more valuable concerning the robot's ability to predict task success, in order to prevent failures.

V. Results and Discussion

A. 2-Stack-Simulation Results

In Fig. 4, we present a comparison of parameter estimates obtained from MLE (blue) and Bayesian estimation, which incorporates priors from our three proposed prior transfer strategies. The Bayesian Estimates

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

TABLE I

ENLISTS THE OBTAINED DISCRETIZATION INTERVALS FOR THE VARIABLES $\mathbf{X}_{\text{STACK}}$ (WHERE $i = 1$ REPRESENTS THE FIRST AND $i = 2$ THE SECOND STACK)AND $\mathbf{X}_{\text{DROP}}$ (IN CM). CONTAINERHEIGHT AND CONTAINERDIAMETER VARIABLES ARE SHOWN FOR EACH CONTAINER SEPARATELY. XOFF/YOFF ARE VALID FOR ALL THREE CONTAINERS. THE COLOR VARIABLE REPRESENTS THE SIMILARLY DISTRIBUTED VARIABLES COLORUP_i, COLORDOWN AND CONTAINERCOLOR

a) 2-Stack-Task: $\mathbf{X}_{\text{Stack}}$			**b) SphereDropping-Task: $\mathbf{X}_{\text{Drop}}$**						
$\texttt{dropOff}_i$	$\texttt{xOff/yOff}_i$	color	xOff/yOff	$\text{height}_{\text{plate}}$	$\text{diameter}_{\text{plate}}$	$\text{height}_{\text{bowl}}$	$\text{diameter}_{\text{bowl}}$	$\text{height}_{\text{glass}}$	$\text{diameter}_{\text{glass}}$
$z_1 : [0.5, 1.4]$	$x_1 : [-3.0, -1.4]$	c_1 : Red	$x/y_1 : [-6.0, -3.6]$	$h_1 : [1.6, 2.0]$	$d_1 : [12, 16]$	$h_1 : [5.3, 6.9]$	$d_1 : [9.6, 12.5]$	$h_1 : [8, 10.4]$	$d_1 : [6, 8]$
$z_2 : (1.4, 2.2]$	$x_2 : (-1.4, -0.4]$	c_2 : Blue	$x/y_2 : (-3.6, -1.2]$	$h_2 : (2.0, 2.5]$	$d_2 : (16, 19]$	$h_2 : (6.9, 8.4]$	$d_2 : (12.5, 15.3]$	$h_2 : (10.4, 12.8]$	$d_2 : [8, 10.2]$
$z_3 : (2.2, 3.2]$	$x_3 : (-0.4, 0.4]$	c_3 : Green	$x/y_3 : (-1.2, 1.2]$	$h_3 : (2.5, 3.0]$	$d_3 : (19, 23]$	$h_3 : (8.4, 10]$	$d_3 : (15.4, 18.2]$	$h_3 : (12.8, 15.2]$	$d_3 : [10.2, 12.2]$
$z_4 : (3.2, 4.6]$	$y_4 : (0.4, 1.4]$	c_4 : Orange	$x/y_4 : (1.2, 3.6]$	$h_4 : (3.0, 3.5]$	$d_4 : (23, 26]$	$h_4 : (10, 11.6]$	$d_4 : (18.2, 21.2]$	$h_4 : (15.2, 17.6]$	$d_4 : [12.2, 14.1]$
$z_5 : (4.6, 1]$	$y_5 : (1.4, 3.0]$		$x/y_5 : (3.6, 6]$	$h_5 : (3.5, 3.0]$	$d_5 : (26, 30]$	$h_5 : (11.6, 13.2]$	$d_5 : (21.2, 24]$	$h_5 : (17.6, 20]$	$d_5 : (14.1, 16]$
Distr.: $\sim \mathcal{N}([0.0, 2])$	Distr.: $\sim \mathcal{N}([0.1, 3])$	Distr.: $\sim \mathcal{U}(1, 4)$	Distr.: $\sim \mathcal{U}([-6, 6])$	Distr.: determined by $\texttt{scalingFactor} \sim \mathcal{U}([0.4, 10])$ (thus, $\texttt{height/diameter}_{\text{plate, bowl, glass}}$ unif. distributed)					

TABLE II

DETAILS THE SETUP OF OUR FIVE TRANSFER SCENARIOS. THE NUMBER OF PARAMETERS CAN BE DETERMINED BY MULTIPLYING THE NUMBER OF INTERVALS FOR EVERY PARENT NODE. E.G., FOR THE 1-STACK TASK WE HAVE $|\texttt{XOFF1}||\texttt{YOFF1}| = 25$ PARAMETERS.

Transfer Scenario	**Data**	**Prior Task T'**	**Target Task T**	**Prior Transfer Methods**
2-Stack-Simulation	$\lvert D' \rvert = 100,000$ $\lvert D_{\text{Train}} \rvert = 20,000$ $\lvert D_{\text{GroundTruth}} \rvert = 880,000$	1-Stack-Simulation $P(\text{onTop1} = 1 \lvert \text{Pa}_{\text{onTop1}}, \mathcal{D}')$ 25 parameters ($\boldsymbol{\theta}'_{\texttt{onTop1}=1 \lvert \boldsymbol{u}'}$ with $\boldsymbol{u}' \in Val(\text{Pa}_{\texttt{onTop1}})$)	2-Stack-Simulation $P(\text{onTop2} = 1 \lvert \text{Pa}_{\text{onTop2}}, \mathcal{D})$ 625 parameters	Direct-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff1,yOff1}\}}$ Cross-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff2,yOff2}\}}$ with $\boldsymbol{u}' \in Val(\{\texttt{xOff1}, \texttt{yOff1}\})$ and Minimal-Repl.
2-Stack-Reality	$\lvert D' \rvert = 135$ (15 per param.) $\lvert D_{\text{Train}} \rvert = 45$ (5 per param.) $\lvert D_{\text{GroundTruth}} \rvert = 180$ (20 per param.)	1-Stack-Reality $P(\text{onTop1} = 1 \lvert \text{Pa}_{\text{onTop1}}, \mathcal{D}')$ 3 parameters ($\boldsymbol{\theta}'_{\texttt{onTop1}=1 \lvert \boldsymbol{u}'}$ with $\boldsymbol{u}' \in Val(\text{Pa}_{\texttt{onTop1}})$)	2-Stack-Reality $P(\text{onTop2} = 1 \lvert \text{Pa}_{\text{onTop2}}, \mathcal{D})$ 9 parameters.	Direct-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff1}\}}$ Cross-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff2}\}}$ with $\boldsymbol{u}' \in Val(\{\texttt{xOff1}\})$ Minimal-Repl.
2-Stack-SimToReal	$\lvert D' \rvert = 10,000$ $\lvert D_{\text{Train}} \rvert = 180$ (20 per param.) $\lvert D_{\text{GroundTruth}} \rvert = 180$ (20 per param.)	2-Stack-Simulation $P(\text{onTop2} = 1 \lvert \text{Pa}_{\text{onTop2}}, \mathcal{D}')$ 9 parameters ($\boldsymbol{\theta}'_{\texttt{onTop2}=1 \lvert \boldsymbol{u}'}$ with $\boldsymbol{u}' \in Val(\text{Pa}_{\texttt{onTop2}})$)	2-Stack-Reality $P(\text{onTop2} = 1 \lvert \text{Pa}_{\text{onTop2}}, \mathcal{D})$ 9 parameters	Direct-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff1}\}}$ Cross-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff2}\}}$ with $\boldsymbol{u}' \in Val(\{\texttt{xOff1}\})$ and Minimal-Repl.
SphereDropping	$\lvert D' \rvert = 333,333$ $\lvert D_{\text{Train}} \rvert = 10,000$ $\lvert D_{\text{GroundTruth}} \rvert = 323,333$	Dropping spheres into a bowl $\mathcal{D}' = \mathcal{D}_{\text{Bowl}}$ 125 parameters ($\boldsymbol{\theta}'_{\texttt{inside}=1 \lvert \boldsymbol{u}'}$ with $\boldsymbol{u}' \in Val(\text{Pa}_{\texttt{inside}})$)	Dropping spheres into a plate $T_{\text{Plate}} : \mathcal{D} = \mathcal{D}_{\text{Plate}}$ Dropping spheres into a glass $T_{\text{Glass}} : \mathcal{D} = \mathcal{D}_{\text{Glass}}$ $P(\texttt{inside} = 1 \lvert \text{Pa}_{\text{inside}}, \mathcal{D})$ 125 parameters	Direct-Repl.: $\boldsymbol{u}' = \boldsymbol{u}$
Stack-To-Drop	$\lvert D' \rvert = 1,000,000$ $\lvert D_{\text{Train}} \rvert = 10,000$ $\lvert D_{\text{GroundTruth}} \rvert = 323,333$	1-Stack-Simulation $P(\text{onTop1} = 1 \lvert \text{Pa}_{\text{onTop1}}, \mathcal{D}')$ 25 parameters ($\boldsymbol{\theta}'_{\texttt{onTop1}=1 \lvert \boldsymbol{u}'}$ with $\boldsymbol{u}' \in Val(\text{Pa}_{\texttt{onTop1}})$)	Dropping spheres into a plate $T_{\text{Plate}} : \mathcal{D} = \mathcal{D}_{\text{Plate}}$ Dropping spheres into a glass $T_{\text{Glass}} : \mathcal{D} = \mathcal{D}_{\text{Glass}}$ $P(\texttt{inside} = 1 \lvert \text{Pa}_{\text{inside}}, \mathcal{D})$ 125 parameters	Cross-Repl.: $\boldsymbol{u}' = \boldsymbol{u}_{\{\texttt{xOff1,yOff1}\}}$ with $\boldsymbol{u}' \in Val(\{\texttt{xOff}, \texttt{yOff}\})$

are depicted for three different weight values, namely $w_{\text{prior}} = \{5(\text{green}), 20(\text{red}), 200(\text{yellow})\}$[4]. In terms of Mean Difference metrics, MLE[5] generally converges faster to the ground truth compared to Bayesian Estimates (Fig. 4.b). Furthermore, Bayesian estimates exhibit slower convergence with increasing prior weight, with the green curve ($w_{\text{prior}} = 5$) being closer to the blue curve than the yellow curve ($w_{\text{prior}} = 200$) throughout all transfer approaches. However, when considering the Max Difference metric (Fig. 4.a), Bayesian Estimates display an advantage, especially when using a weight of $w_{\text{prior}} = 5$. This advantage is particularly evident when employing Direct-Replacement or Minimal-Replacement within the initial sample range of up to 7500 samples. Most importantly, however, Bayesian estimation outperforms MLE concerning failure prevention metrics. Even with no new data ($|\mathcal{D}| = 0$), we achieve approximately 70% stacking success for Cross-Replacement and Direct-Replacement, and a remarkable 100% stacking success for the Minimal-Replacement Strategy.

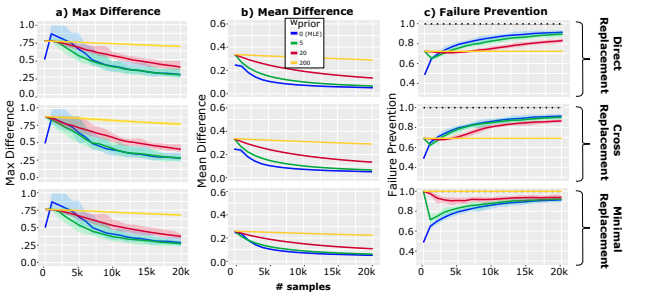

Fig. 4. **2-Stack-Simulation** performance comparison between MLE ($w_{\text{prior}} = 0$) and Bayesian Estimation with $w_{\text{prior}} = \{5, 20, 200\}$. The dotted black line in the Failure Prevention plots reports the action success after performing failure prevention based on the ground truth parameters.

B. 2-Stack-Reality and 2-Stack-SimToReal Results

The task success difference between simulation and reality is around 15% for the 1-Stack task and 31% for the 2-Stack task (see Fig. 5). Nevertheless, we observe that the real

[4]These values were chosen to represent small, medium and large weights. To ensure that prior and new data are equally significant, we would need a dataset of size $|\mathcal{D}| = [3125; 12,500; 125,000]$ respectively.

[5]MLE does not incorporate any prior information. Therefore, if $M[\boldsymbol{u}] = 0$, the division in Eq.3 becomes undefined. We then set $\theta = 0.5$, which was found to yield the best results among $\theta = \{0, 0.5, 1\}$.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

stacking success follows a similar trend as in simulation: as the x-Offsets grow, the likelihood of stacking success decreases.

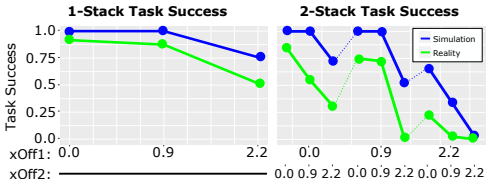

Fig. 5. Stacking task success in simulation and reality.

Next, we discuss the outcome of the 2-Stack-Reality experiments as shown in Fig. 6. While the parameter estimates show a larger variance due to the relatively small size of training data ($|\mathcal{D}'| = 135$, $|\mathcal{D}| = 45$) and ground truth data-set (180) samples, the results confirm the findings of the 2-Stack-Simulation (Sec. V-A) scenario.

We further compare the two investigated priors, namely $T'_{\text{1-Stack-Reality}}$ and $T'_{\text{2-Stack-Simulation}}$ in more detail. This comparison is particularly interesting as with the $T'_{\text{1-Stack-Reality}}$ prior, we face the challenge of transferring BN parameters from a simpler to a more complex task, and with the $T'_{\text{2-Stack-Simulation}}$ prior, we encounter the sim-2-real transfer problem. Interestingly, we observe that the difference in parameter estimates for the two priors is not substantial, especially considering the Minimal-Replacement strategy for obtaining the $T'_{\text{1-Stack-Reality}}$ prior parameters. While it took around 17 minutes to collect the 1-Stack-Reality prior with $|\mathcal{D}'| = 135$, the 10,000 samples for the 2-Stack-Simulation prior were gathered in just 2 minutes. However, it is also essential to consider that building a simulation environment with properly tuned physics parameters can significantly exceed the time required for obtaining the 1-Stack-Reality prior (17 minutes). Consequently, our proposed prior initialization strategies prove highly valuable in enabling robots to learn BNs and their parameters from real robot experiments while also leveraging their prior task experience, such as the 1-Stack task, in more complex situations, like the 2-Stack task.

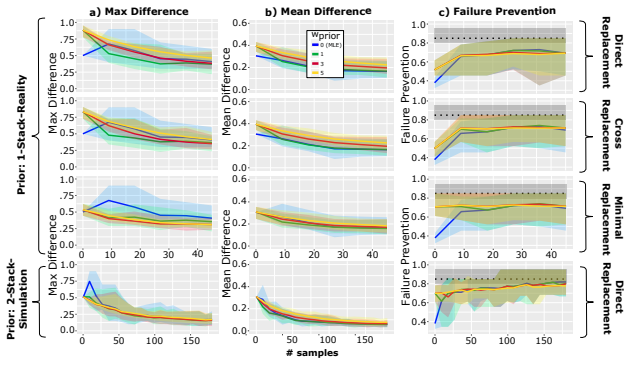

Fig. 6. **2-Stack-Reality** performance comparison between MLE ($w_{\text{prior}} = 0$) and Bayesian estimation with ($w_{\text{prior}} = \{1, 3, 5\}$).

C. SphereDropping Results

The results obtained from the DroppingSphere-Simulation scenario (Fig. 7) confirm the findings from the 2-Stack-Simulation (Sec.V-A) and 2-Stack-Reality experiments (Sec.V-B). A notable distinction is observed for T_{Plate}, where the prior parameters from T'_{Bowl} are already quite close to the parameters of T_{Plate}. This is evident from the Mean Difference plot in Fig. 7-b. In this case, Bayesian estimates show better proximity to the ground truth than MLE and thus outperform MLE concerning the Max Difference metric. Moreover, the Bayesian estimates based on the priors obtained from our approach achieve almost the maximum task success rate, with only a 4% difference (see Fig. 7-c) compared to the action success achieved when employing the ground truth parameters for failure prevention actions (indicated by the black dotted line in the Failure Prevention plots).

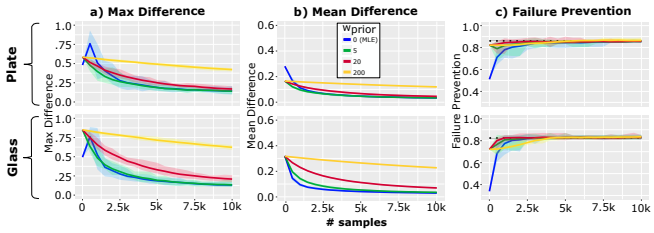

Fig. 7. **SphereDropping-Simulation** performance comparison between MLE ($w_{\text{prior}} = 0$) and Bayesian estimation with $w_{\text{prior}} = \{5, 20, 200\}$.

D. Stack-To-Drop Results

The results from the Stack-To-Drop transfer scenario (Fig. 8) align with findings from the other experiments. In particular, compared to MLE, our transferred priors yield a 20% improvement in failure prevention when $|D| = 0$ (see Fig. 8-c). However, as data size $|D|$ increases, MLE catches up with the Bayesian approach more rapidly than in previous scenarios, reducing the required data size for MLE to be equally effective in preventing failures. Using priors from dropping spheres into a bowl, it takes 2,600 samples (Fig. 7-c), while using stacking cubes as priors, it takes 315 samples to achieve similar failure prevention capabilities for the task of dropping a sphere into a glass (Fig. 8-c).

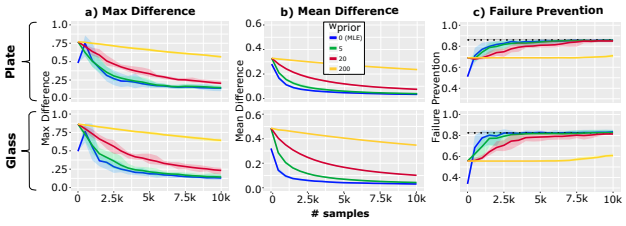

Fig. 8. **Stack-To-Drop** performance comparison between MLE ($w_{\text{prior}} = 0$) and Bayesian estimation with $w_{\text{prior}} = \{5, 20, 200\}$.

E. Discussion

In this work, we propose three strategies that utilize semantic similarity and relatedness between the variables of two CBNs to generate and transfer informed CBN distribution priors. We show that these strategies are effective across various transfer tasks, including sim-2-real, transferring parameters to more complex tasks with a larger number of parameters and even between two different tasks. Comparing our three strategies, we consistently find that the Minimal-Replacement strategy, when applicable, performs the best.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.

This is due to its conservative nature in assigning large parameter values. For instance, in the 2-Stack experiments, the Minimal-Replacement strategy assigns high success chances only to x/yOff1 and x/yOff2 combinations near the cube center. In contrast, the Direct-Replacement strategy assigns high success predictions to central x/yOff1 intervals even for x/yOff2 intervals located far from the cube center.

Next, we discuss the role of w_{prior} in our experiments. Notably, when using large values for w_{prior} (e.g., 200 in the case of 2-Stack-Simulation), we observe a significant reduction in the Mean Difference convergence rate across all prior transfer strategies and tasks. With $w_{\text{prior}} = 200$, the prior data carries equal importance to 200 new data points. As a result, we would need an extensive dataset of at least $125,000$ samples to ensure that the prior and new data are equally significant for all 625 parameters of the 2-Stack task. Surprisingly, large w_{prior} do not necessarily lead to better failure prevention results, even when the prior parameters closely match the ground truth (e.g., SphereDropping-Simulation-Bowl in Fig. 7). Generally, we observed the best results with smaller w_{prior} values, such as 5 (in the case of 2-Stack-Simulation and SphereDropping-Simulation) or 1 (in the case of 2-Stack-Reality). We conclude that an essential factor for the choice of w_{prior} should be the expected size of new task samples $|\mathcal{D}|$. For instance, in the 2-Stack-Reality scenario, we generally opt for smaller weight values due to the smaller number of collected task instances (180 training samples) compared to the 2-Stack-Simulation scenario ($20,000$ training samples). In future work, we aim to investigate more automated strategies for selecting w_{prior}. One approach could involve utilizing the notion of task similarity between T and T' to obtain a measure of trust in the prior, which could be leveraged to adapt w_{prior} more effectively.

To summarize, we show that even though ML estimates converge faster toward the ground truth, the Bayesian estimates, which incorporate the priors from our proposed transfer methods, prove significantly more valuable in real decision-making scenarios, particularly when they are used as input for preventing task execution failures. As a result, our transferred priors allow robots to commit between 17.5% (2-Stack-Reality + 1-Stack-Reality prior + Cross/Direct-Replacement), 20% (Drop-To-Sphere) and 50% (2-Stack-Simulation + Minimal-Replacement) fewer task execution failures when no new target task experiments are available.

VI. Conclusion

In this paper, we investigate the challenging problem of applying prior experience in novel robot tasks, particularly focusing on the transfer of CBN probability distribution parameters. We propose three strategies to transfer parameter estimates between two CBNs based on the semantic similarity of their variables and show that they can help robots improve their action success in novel situations by up to 50%. The proposed prior transfer strategies are, therefore, an important step to increase the applicability of Bayesian in real-world robotic applications and improve the robot's capability to utilize prior experience zero-shot in novel situations.

References

[1] A. S. Bauer, P. Schmaus, F. Stulp, and D. Leidner, "Probabilistic effect prediction through semantic augmentation and physical simulation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[2] A. Mitrevsk, P. G. Plöger, and G. Lakemeyer, "Ontology-assisted generalisation of robot action execution knowledge," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[3] K. Xu, E. Ratner, A. Dragan, S. Levine, and C. Finn, "Learning a prior over intent via meta-inverse reinforcement learning," in *Proceedings of the 36th International Conference on Machine Learning*, 2019.

[4] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018.

[5] T. Hellström, "The relevance of causation in robotics: A review, categorization, and analysis," *Paladyn, Journal of Behavioral Robotics*, vol. 12, no. 1, 2021.

[6] J. Brawer, M. Qin, and B. Scassellati, "A causal approach to tool affordance learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[7] M. Diehl and K. Ramirez-Amaro, "Why did i fail? a causal-based method to find explanations for robot failures," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, 2022.

[8] ——, "A causal-based approach to explain, predict and prevent failures in robotic tasks," *Robotics and Autonomous Systems*, vol. 162, 2023.

[9] H. Wang, I. Rish, and S. Ma, "Using sensitivity analysis for selective parameter update in bayesian network learning," *Association for the Advancement of Artificial Intelligence (AAAI)*, 2002.

[10] D. Song, K. Huebner, V. Kyrki, and D. Kragic, "Learning task constraints for robot grasping using graphical models," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010.

[11] A. Niculescu-Mizil and R. Caruana, "Inductive transfer for bayesian network structure learning," in *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, M. Meila and X. Shen, Eds., vol. 2. San Juan, Puerto Rico: PMLR, 21–24 Mar 2007.

[12] V. Rodríguez-López and L. E. Sucar, "Knowledge transfer for causal discovery," *International Journal of Approximate Reasoning*, vol. 143, 2022.

[13] Y. Hou, A. Yang, W. Guo, E. Zheng, Q. Xiao, Z. Guo, and Z. Huang, "Bearing fault diagnosis under small data set condition: A bayesian network method with transfer learning for parameter estimation," *IEEE Access*, vol. 10, 2022.

[14] Y. Zhou, T. M. Hospedales, and N. Fenton, "When and where to transfer for bayesian network parameter learning," *Expert Systems with Applications*, vol. 55, pp. 361–373, 2016.

[15] X. Ru, X. Gao, Y. Wang, and X. Liu, "Bayesian network parameter learning using constraint-based data extension method," *Applied Intelligence*, vol. 53, no. 9, p. 9958–9977, aug 2022.

[16] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques - Adaptive Computation and Machine Learning*. The MIT Press, 2009.

[17] M. Scutari, "Learning bayesian networks with the bnlearn R package," *Journal of Statistical Software*, vol. 35, no. 3, 2010.

[18] D. Colombo and M. H. Maathuis, "Order-independent constraint-based causal structure learning," *Journal of Machine Learning Research*, vol. 15, no. 116, 2014.

[19] R. Nagarajan and M. Scutari, *Bayesian Networks in R with Applications in Systems Biology*. New York: Springer, 2013, iSBN 978-1-4614-6445-7, 978-1-4614-6446-4.

[20] M. J. Vowels, N. C. Camgoz, and R. Bowden, "D'ya like dags? a survey on structure learning and causal discovery," *ACM Comput. Surv.*, mar 2022.

[21] Y.-C. Chen, T. A. Wheeler, and M. J. Kochenderfer, "Learning discrete bayesian networks from continuous data," *J. Artif. Int. Res.*, vol. 59, no. 1, may 2017.

[22] Z. Ji, Q. Xia, and G. Meng, "A review of parameter learning methods in bayesian network," in *Advanced Intelligent Computing Theories and Applications*. Springer International Publishing, 2015.

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2024, Yokohama, Japan. Cite as RA-L paper.