Combining Fairness and Causal Graphs to Advance Both

Lea Cohausz[1,*], Jakob Kappenberger[1] and Heiner Stuckenschmidt[1]

[1]University of Mannheim, Germany

Abstract

Recent work on fairness in Machine Learning (ML) demonstrated that it is important to know the causal relationships among variables to decide whether a sensitive variable may have a problematic influence on the prediction and what fairness metric and potential bias mitigation strategy to use. These causal relationships can best be represented by Directed Acyclic Graphs (DAGs). However, so far, there is no clear classification of different causal structures containing sensitive variables in these DAGs. This paper's first contribution is classifying the structures into four classes, each with different implications for fairness. However, we first need to learn the DAGs to uncover these structures. Structure learning algorithms exist but currently do not make systematic use of the background knowledge we have when considering fairness in ML, although the background knowledge could increase the correctness of the DAGs. Therefore, the second contribution is an adaptation of the structure learning methods. This is evaluated in the paper, demonstrating that the adaptation increases correctness. The two contributions of this paper are implemented in our publicly available Python package *causalfair*, allowing everyone to evaluate which relationships in the data might become problematic when applying ML.

Keywords

Fairness, Causal Models, Algorithmic Bias, Bayesian Network Structure Learning

1. Introduction

The importance of fairness in AI and, more specifically, Machine Learning (ML) has been recognized in recent years, in particular in areas directly concerning humans, such as education, finance, or health care [1, 2, 3]. One way to discuss whether an ML system can be considered fair is to look at the outcome of the ML model [4]. Then, fairness is usually evaluated by looking at metrics of algorithmic bias[1], such as Demographic Parity (DP) [4]. These encode different notions of fairness, and once a metric has been decided on, it can indicate whether a model is fair according to this notion. However, apart from the normative[2], overarching question of which metric is most fair, the choice of metric, its interpretation, and how we should deal with potential fairness concerns is context-dependent. This aspect was also noted in previous works, remarking that taking a causal view allows us to account for much of the context-dependency and, hence, to properly assess whether an AI system's outcome should be considered fair [5, 6, 7].

AEQUITAS 2024: Workshop on Fairness and Bias in AI | co-located with ECAI 2024, Santiago de Compostela, Spain

*Corresponding author.

lea.cohausz@uni-mannheim.de (L. Cohausz); jakob.kappenberger@uni-mannheim.de (J. Kappenberger); heiner.stuckenschmidt@uni-mannheim.de (H. Stuckenschmidt)

[1]Although frequently coined fairness metrics, we stick to the term metrics of algorithmic bias to indicate that fairness is a multi-dimensional concept.

[2]That means that the decision depends on a person's individual beliefs.

In particular, Chiappa et al. argued for using Causal Bayesian Networks (CBNs) to understand how variables influence each other and what the data-generating mechanism looks like [5]. This procedure can help determine which metrics of algorithmic bias to choose, how to interpret them, and how to deal with potential fairness concerns.

However, so far, no clear classification of specific causal structures and their indications for fairness exists that also considers that the data is used in an ML context. What follows is that there is no implementation that can automatically detect different kinds of structures, which allows researchers and practitioners to check what parts of their data they intend to learn a model on could be problematic. In addition, existing work assumes that the CBNs are already constructed. However, constructing CBNs is non-trivial as we typically do not know all relationships existing in the data (i.e., cannot simply use expert knowledge), and data-driven causal structure learning methods are known to be error-prone in more complicated settings [8, 9]. In many fairness settings, however, we automatically have some background knowledge, i.e., we know which variables in the data are sensitive, and this has certain implications for learning the causal structure, as we will see. Hence, the contributions of the paper are twofold.

- We create a classification of different causal structures and explain their implications for fairness assessment.
- We adapt data-driven causal structure learning algorithms to include background knowledge we have in fairness settings. We also evaluate whether the adaptation increases the accuracy of the CBNs on synthetic data for which we know the ground truth.[3]

We implemented both contributions in our publicly available Python package *causalfair* that researchers and practitioners can use to assess whether and how the data used for Machine Learning (ML) is problematic. The package can be found online (https://github.com/lea-cohausz/causalfair).[4]

2. Background

Before discussing these aspects, we will briefly detail what CBNs are. The graphical part of CBNs consists of Directed Acyclic Graphs (DAGs). A DAG is a graph with nodes (also called vertices) $\mathcal{X}$ that, in the case of a Bayesian Network, encode random variables and directed edges $\mathcal{E}$ connecting the vertices [10]. An edge from one node to another, i.e., $x_i \rightarrow x_j$, means that the first node causally influences the second node. A path in a DAG encompasses a sequence of directed edges, i.e., $x_i \rightarrow x_j \rightarrow ... \rightarrow x_t$. Furthermore, it holds for a CBN that a variable x_i is only dependent on its parents and independent of all other variables given its parents, i.e.:

$$P(x_i) = \prod P(x_i|Pa(x_i)) \tag{1}$$

where $Pa(x_i)$ are the parents of x_i. Therefore, CBNs encode independence information. In DAG (6) in Figure 1, A and Y are conditionally independent given X, which we write as $A \perp Y|X$. This is equivalent to saying that all information relevant for Y is encoded in X, and we do

[3] In addition, our online repository contains an example of a real-life dataset.
[4] Our experiments are also available here https://github.com/lea-cohausz/Causalfair_Experiments.

not need to know A to learn something about Y [10]. Note, however, that A and Y are only conditionally independent, which means that the variables are correlated. An imperfect ML model may use this correlation, even though all information necessary is encoded in X.

2.1. Sensitive Variables

When assessing whether an ML model is fair or not concerning the model's outcome, we usually use metrics of algorithmic bias [4]. All of these metrics have in common that they monitor differences in the model's outcome with regard to sensitive variables. These sensitive variables are usually demographic variables [4, 11]. Demographic variables are, among others, variables such as gender, age, socio-economic status, and variables pertaining to this information. Another definition is that demographic features are features that cannot be changed within the context of the setting [11]. For example, if we have a model in the educational setting, all those variables should be considered demographic and potentially sensitive, which cannot be changed within the educational setting (i.e., gender is not changed by education, but educational attainment itself is). The different fairness metrics require different absences of statistical relationships between a sensitive feature and the prediction of the target variable.

Because DAGs encode independence relationships and information on which variables influence each other, they are well suited for uncovering potential fairness problems in the data. Chiappa and Isaac showed that by looking at DAGs representing the data-generating mechanism, we could determine whether sensitive variables causally influence the target variable or not [5]. Based on this, we can then make an informed decision about whether this is actually problematic and which fairness metric can be used [5]. However, no clear classification of different structures was made, and existing work mostly focused on whether a structure is potentially problematic according to the causal structure [5, 7]. However, when we have the ultimate goal of using the data to build ML models, more considerations apply [6]. Most importantly, ML models frequently use correlated but causally unconnected variables, even if the information in this variable is also entailed in another causally connected variable [12]. To the best of our knowledge, we are the first to provide a clear classification of the structures in which sensitive variables are involved.

3. Different Types of Structures

Figure 1 shows different structures, including a sensitive and a target variable we identified as potentially existing within a DAG. These structures (i.e., the different ways in which sensitive features and the target can be part of a larger DAG) can further be classified with respect to whether and how the sensitive variable involved in the structure is problematic. We have identified four classes of structures regarding this. We want to highlight again that the final decision of whether a sensitive variable has a problematic influence is still up to the expert.[5] In general, we speak of a problem for ML if it is likely that an ML model will use the sensitive variable or variables heavily dependent on observed or unobserved sensitive variables (proxies)

[5]We recommend Cohausz et al. to receive an idea of when we might deem relationships problematic and how this relates to selecting relevant fairness metrics [7].

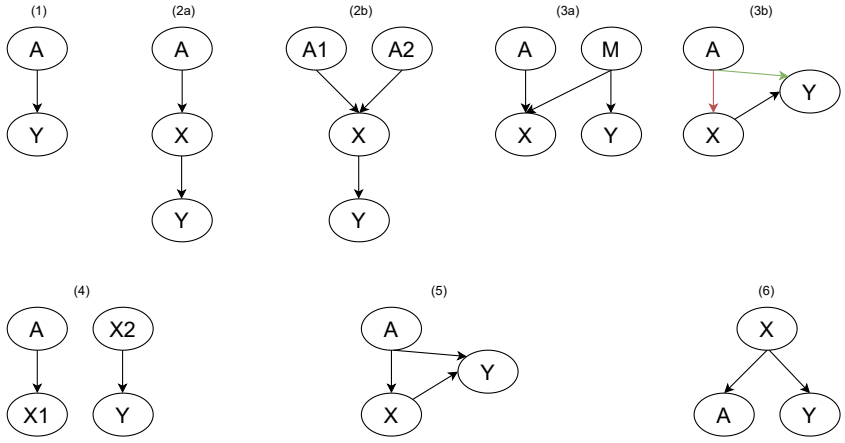


Figure 1: Different causal structures. The numbers correspond to the numbers in the table.

for its prediction. We will now briefly mention these classes (i.e., the different ways in which sensitive variables have or do not have a potential impact on the target and, thus, fairness) before delving into the different structures within these classes. In the following, we use A to refer to a sensitive variable and Y to refer to the target; other letters refer to other predictive variables.

- Potentially problematic structures that are problematic for ML (structures 1, 2a, 2b, 5). This class is characterized by a direct or indirect connection or both but with the same direction of effects. To deal with the fairness problem, we need to remove A and potentially mitigate the effect of A on X if the relationships are deemed problematic. In the following, we call these **problematic variables**.
- Unproblematic structures that are unproblematic for ML (structure 4). This class is characterized by no undirected path between A and Y. In the following, we call these **unproblematic variables**.
- Unproblematic structures but potentially problematic from an ML perspective (structure 3a). This class occurs if all directed paths from A to Y are blocked. In the following, we call these **blocked variables**.
- Potentially problematic structures where removing the sensitive variable (structure 3b) is problematic. This class occurs if there is a direct and indirect connection from A to Y and the effects are opposing. In the following, we call these **opposing effects variables**.

causalfair returns both the exact structures a sensitive variable is involved in as well as the classification of these structures. We want to highlight that the different structures within the same class still have to be viewed and handled differently [7]. We will now discuss the different classes in slightly greater detail. Table 1 summarizes the structures.

Problematic Variables: As already mentioned, structures (1), (2a), (2b), and (5) are problematic. They all have in common that there is at least one directed path from the sensitive

variable to the target. Information about the target is encoded in the sensitive variables, which means that an ML model might use the correlation and, thus, place direct importance on the sensitive variable – which is potentially problematic. In the indirect case, as all information is also encoded in the mediating variable, the model may or may not place importance on it. Still, information from this variable will be passed on through the mediating variable. If we do not think that information from the sensitive variable should be used, then we should remove the variable and, in the indirect case, mitigate the effect of the variable while monitoring fairness metrics. We may also decide that only the direct effect should not be used (e.g., in (5)). *Example:* As an example, ethnicity may influence students' grades in a specific course due to discrimination. In this case, the sensitive variables influence the target in such a way that the target is also biased.

Unproblematic Variables: Structure (4) stands for all networks that are fragmented without a connection between the subnetwork containing the sensitive variable and the subnetwork containing the target variable. In these cases, we do not have to be worried much. No information about Y is encoded in A. Still, as imperfect ML models (in particular Neural Networks) tend to assign some importance even to irrelevant features [13], it is probably best to remove both A and $X1$. Then, nothing needs to be monitored or mitigated. *Example:* Gender may influence height, but neither of those variables is relevant to whether students pass a course. Hence, there is no statistical relationship between any of the variables of the different fragments at all.

Blocked Variables: For (3a), similar to (4), there is no path of directed edges that leads from the demographic variable to the target. In difference to (4), here, there is no lack of connection. Instead, M blocks the paths, meaning no information from the sensitive variable is transported to the target. From a network perspective, the structure would be unproblematic, but from an ML perspective, it might not be. X, which is influenced by A, is correlated with Y. Although all information regarding the target is contained in M and $X \perp Y|M$, an ML model may still use and assign importance to X due to the correlation. If the model uses X, it will likely also use A to correct for the bias in X. This consequence was also observed by Ashurst et al. [6]. Therefore, we have two options for handling this. We can remove both A and X, but if we remove A, we must also remove X. Otherwise, we would introduce a bias in our predictions that is not reflected by the real and unbiased target variable. Alternatively, we leave both in the data and closely monitor all metrics for algorithmic bias. *Example:* The students' motivation influences both passing course X and passing course Y (the target). In course X, the professor discriminates against one gender. In this case, all information relevant to the target is encoded in the motivation, and X and gender are statistically independent of the target. However, course X is not independent of the motivation, which is a relevant variable for the target. This relationship may lead to an ML model that places weight on course X and, consequently, gender.

Opposing Effects Variables: Although (3b) looks like (5) and, thus, should be classified as problematic, this becomes a very different case when the direct and indirect effects are opposing. This is the case if we have a missing variable. For example, if M in (3a) is not observed, then the DAG learned from data will be (3b). If we do not know M, then it will appear like X and A influence Y, and A also influences X. The influence A has on X corrects itself through the connection $A \rightarrow Y$, meaning the target is unbiased. From a causal structure perspective, such a structure is clearly problematic. However, from an ML perspective, we again have the two options we had for (3a). We either leave X and A in, as the opposing effects of A on X and Y,

Table 1
This table provides a summary of whether the structures in the corresponding figure are problematic from a causal structure and ML point of view, and what strategy to use to mitigate algorithmic bias.

structure	(1)	(2a)	(2b)	(3a)	(3b)	(4)	(5)	(6)
problematic according to causal structure	yes	yes	yes	no	yes	no	yes	no
problematic for ML	yes	yes	yes	maybe	maybe	no	yes	yes
strategy	remove A	remove A, mitigate influence of A on X	remove As, mitigate influence of As on X	remove A & X or none	remove A & X or none	none, or remove X1 and A	remove A, mitigate influence of A on X	remove A

respectively, may cancel each other out. Or we must remove both. Then, however, we may lose a lot of predictive power. Although (5) and (3b) look identical from a network perspective, the implications are very different. Hence, for *causalfair*, we check when such a structure exists whether the effects of A on X and of A on Y point in opposite directions (demonstrated in the graph with the two colors). If this is not the case, then it is (5). Otherwise, *causalfair* informs the user of the structure. It is important to know that the resulting graph learned from data is not strictly speaking a causal graph as a relevant variable is missing. *Example:* If the variable motivation, as described in the example for the blocked variables, is unobserved, the graph (3b) would likely be learned by a structure learning algorithm.

Finally, structure (6) has been considered in the literature before, but we argue that we usually do not have to think about this case because sensitive variables are – at least according to the definition explained above – usually not changeable by other variables.[6]

4. Causal Structure Learning

Detecting the above-described structures relies on accurate DAGs. These DAGs first need to be constructed. There are several ways to do so:

1. **Expert knowledge.** While we can construct the DAG using background knowledge [14], we usually do not know about causal relationships, or our ideas might not match the data. Still, expert knowledge is important: We often know about the temporal ordering of variables and, therefore, know that certain relationships cannot exist (e.g., grades cannot influence ethnicity).
2. **Data-driven methods.** Research on causal structure learning has produced several methods to learn CBNs from data [15]. If certain assumptions hold and data is sufficient, these methods work rather well [8]. In more realistic cases, however, the methods cannot reliably produce accurate DAGs [9].
3. **Combining expert knowledge and data-driven methods.** We may know that some relationships in the data are impossible or must exist, but we do not know about all relationships. We can feed this knowledge to the structure learning algorithms. Although

[6] If, however, this happens to be false in a specific setting, then we should remove A to avoid that an ML model uses the correlation.

combining both methods seems to lead to better results, doing so has been researched comparatively poorly, and some data-driven methods do not even allow the incorporation of background knowledge [16]. In part, this lack of research is because there is no general procedure for it, and it greatly depends on the data, knowledge, and general situation.

We argue, however, that when constructing a graph to assess fairness, we can use a standard procedure to combine background knowledge and data-driven methods. The reason for this is the background information we automatically have when considering fairness.

4.1. Background Information

As mentioned in section 2, it follows from a definition of sensitive features that non-sensitive variables cannot influence them [11]. Additionally, the target variable usually follows all other variables temporally. For example, if we try to predict admission to a university, all information that can be used has existed longer than the admission decision. Therefore, we can separate the variables into three groups: target variables (which cannot influence any other variables), sensitive variables (which cannot be influenced by any other variables), and regular predictive variables, for which it logically follows that they cannot influence sensitive variables or be influenced by the target. There may also be situations where sensitive variables can be influenced by other sensitive variables or where we know there is an order within the other predictive variables. However, we generally have at least three tiers: the target, other predictive features, and sensitive variables. With the specification of these tiers, we already have a lot of background knowledge: we can require that the data-driven structure learning methods do not include any edges that are impossible according to this specification. Using this background knowledge is particularly helpful, as it is also the knowledge we need to evaluate algorithmic bias, anyhow: knowing which variables are sensitive and what the target is. Additional knowledge we have about the structures can also be specified.[7]

When we now want to learn DAGs from data, we first need to choose among the families of data-driven methods. We will evaluate one method from each of the three most popular families: constraint-based methods, score-based methods, or methods from functional causal modeling and discuss how the background knowledge can be used [15].[8]

4.2. Constraint-Based Structure Learning

Constraint-based structure learning consists of two stages. During the first stage, edges are removed iteratively from an initially complete undirected graph by performing independence tests [15]. Edges can be removed when two variables are (conditionally) independent of each other. Whenever an edge is removed, the variables that make these variables conditionally independent are stored. For example, if A and B are independent given C, i.e., $A \perp B|C$, then C is stored. During the second stage, as many edges as possible are oriented. To do this, we look at groups of three variables A, B, C, and their separating sets. If we have two variables

[7] That is, whether certain variables cannot have ingoing edges or cannot be influenced by certain other variables.

[8] Hybrid methods connecting constraint-based and score-based structure learning also exist [15]. In practice, hybrid methods have been proven to work less well than the mentioned individual methods [9, 8]. Hence, we will not consider them here.

A, B that are conditionally independent and both are dependent on the same third variable C and their separating set does not include C, i.e., $C \notin S_{AB}$, then we have that $A \rightarrow C$ and $B \rightarrow C$. C is a so-called collider, and A, B, C form a v-structure. After all v-structures are identified and the corresponding edges are oriented, other edges are oriented to avoid new v-structures. This concludes the second stage. It has to be noted that not all edges are usually oriented, as only those edges that are part of a v-structure or directly avoid a v-structure can be oriented. Therefore, constraint-based methods do not return a DAG but a Complete Partial DAG (CPDAG). Constraint-based methods are guaranteed to return the correct CPDAG if the independence tests return correct results [15, 10]. Constraint-based algorithms are known to miss more edges than other methods but also insert fewer incorrect edges [9, 8]. We use the PC-Stable (abbreviated in this paper as PC) algorithm, which has been found to work well [15].

Adaptation: Including background information in constraint-based methods is not straightforward as the first stage cannot really be modified, and no implementation so far allows a user to specify background information [16]. Our approach is to use the background information at the end of the second stage: If we have an undirected edge and our background knowledge does not allow one direction, then the edge is oriented accordingly. Afterward, further edges are oriented to avoid v-structures again. Compared to the adaptations of the other methods, this method makes comparatively little use of the background information. It is also not guaranteed that relationships that go against our background knowledge do not exist because the edge may already have been oriented during the v-structure orientation. However, if the CPDAG is correct until we inject the background knowledge, the resulting graph will also be correct.

4.3. Score-Based Structure Learning

In score-based structure learning, we aim to find a DAG that maximizes a score [15]. Hence, the search space of possible graphs must be searched, and the possible graphs must be compared with a score (e.g., an information-theoretic score). Searching the space of possible graphs is usually (though not always) done with a heuristic approach. Despite its simplicity, one algorithm frequently used directly or in some variants is the Hill-Climber (HC) [9, 8]. HC starts with an empty graph and iteratively adds or deletes those edges that lead to the highest increase in the chosen score until the score no longer improves. A DAG that is at least a local maxima is returned, but reaching the global maxima is not guaranteed [15].

Adaptation: Adapting score-based methods to handle the background information is easier, as we can restrict the search space, i.e., edges that are impossible according to our classification will never be added [16].[9] Constaninou et al. recently experimented with different kinds of background knowledge and their effect on the accuracy of DAGs but found that restricting edges only has a small effect [16]. However, we limit the search space more fundamentally.

4.4. Functional Causal Models

The key idea of Functional Causal Modeling is that variables can be determined by a function of their parent variables and a noise term that is independent of their parents. If the function is

[9] Similarly, we could also add information that a certain edge needs to exist – then the edge is directly added and can never be removed.

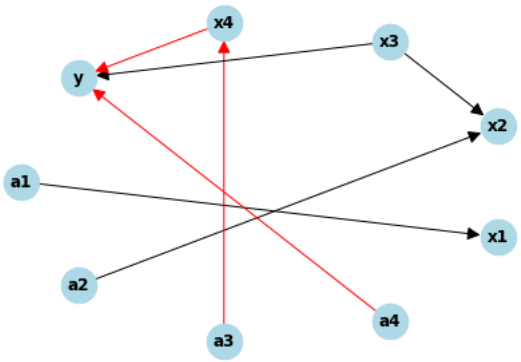


Figure 2: The smallest DAG we created to evaluate against. The red paths are paths where problematic information is transported to the target y.

correctly identified, it is the case that the noise term is only independent of the parent variables for one direction and not for the other. Hence, algorithms belonging to this family search for such relationships between variables. It should be noted that this method is usually used for continuous data, although it can also be used for discrete data. One of the most prominent algorithms belonging to this family is the Linear Non-Gaussian Acyclic Model (LiNGAM) [17].

Adaptation: Similar to HC, we can include background knowledge by preventing LiNGAM from considering certain relationships. Those are then not even attempted to be modeled.

5. Evaluation

We will now evaluate whether our adaptations increase the correctness with which DAGs are learned. For this, we will look at several ground truth DAGs from which we sample data. Then, we will attempt to reconstruct the DAGs. We will vary the data size and the background information available. Additionally, we will check whether the sensitive variables are correctly classified according to the different classes we defined in section 3. We have the following research questions:
RQ1: Does using background information improve the correctness of the models?
RQ2: Is background information particularly helpful for specific data sizes or methods (PC-Stable, HC, LiNGAM)?
RQ3: Does the classification accuracy of demographic variables according to section 3 increase with more background information correspondingly?

5.1. Strategy

In order to evaluate the research questions, we need to know the ground truth DAGs. For this, we selected five Bayesian Networks (BN) from the "bnlearn" library that are frequently used to evaluate structure learning algorithms: asia, earthquake, sachs, alarm, and insurance [18]. For each of the DAGs, we selected some root nodes to represent the sensitive variables. We also selected one of the leaf nodes as the target. Because sampling from these BNs produces discrete

data, but we also want to test with continuous data, we created three additional synthetic DAGs of different sizes for which we sampled continuous data. An example of the smallest continuous network we created can be seen in Figure 2. We specified non-linear relationships for two of the networks (II, III). A summary of the ground truth DAGs can be seen in Table 2.

For each of the networks, we extracted which variables belong to each of the four classes of sensitive variables. For Figure 2, we have that $a3$ and $a4$ belong to the class that is problematic regarding both the causal structure and from the ML perspective (the paths they are involved in are highlighted in red). For $a1$, we have that it is neither, as it has no connection to y. $a2$ is not a problematic variable, as $x2$ blocks it, but it might be problematic when using ML. In this DAG, there is no opposing effects variable setting.

Having gathered the ground truth information, we can now run the experiments. For each DAG, we vary the number of data instances used (500, 1000, and 10000). We also vary whether we have information available (Info) or not (No Info). For each configuration, we sample the data 30 times to receive reliable results. In detail, we proceed like this:

Algorithm 1 The algorithm shows the setup of the experiments for each DAG.

```
for sample size in {500, 1000, 10000} do
    for experiment in range(0,30) do
        sample data
        for method in {PC, HC, LiNGAM} do
            for information in {No Info, Info} do
                learn DAG
                compare to ground truth
```

We compare the correctness of the graph by a) computing how many of the true edges are present in the computed DAG[10] (true positives) and b) how many edges in the computed DAG are incorrect (false positives).[11] To make the values comparable across DAGs, we normalize them by dividing them by the number of actually existing edges in the ground-truth DAG. This procedure means that the range for the incorrect edges is theoretically $[0, \infty]$, as, of course, more incorrect edges can be inserted than correct edges exist. Still, this normalizes the value with regard to the size of the DAG, and in practice, the value is never larger than 1. For the true positives, the value is bounded to 1. Likewise, for *RQ3*, we look at the accuracy with which variables are classified into the four classes.

5.2. Results

Figure 3 (a) shows the results relevant for answering *RQ1*. We can see that providing information increases the correctly found edges compared to having no information available. There are much fewer wrongly placed edges when using information compared to not using information.

[10] Note that for PC, we evaluate against the CPDAG and that the ground-truth CPDAG also changes with more information available. Only exact matches (i.e., same orientation or both unoriented) are counted.

[11] Note that while these are usual metrics in structure learning research, other metrics are also frequently used, such as, e.g., the Hamming Distance [9, 15]. However, we believe this provides a relatively easy-to-understand view of the results.

Table 2
This table provides a summary of the DAGs used in the evaluation. It states the number of nodes and edges, problematic variables, blocked variables, opposite effects variables, and unproblematic variables. The final column shows the average percentage of correct edges found when using the structure learning algorithms across all settings.

Name	|Nodes|	|Edges|	|Problematic|	|Blocked|	|Opposing Effects|	|Unproblematic|	% correct
asia	8	8	2	0	0	0	0.66
earthquake	5	4	2	0	0	0	0.93
sachs	11	17	1	0	0	1	0.44
alarm	37	46	9	0	2	0	0.66
insurance	27	52	1	0	0	0	0.60
Synthetic I	9	7	2	1	0	1	0.89
Synthetic II	10	13	4	0	2	0	0.65
Synthetic III	20	29	3	2	0	1	0.48

That this metric is more affected than the one measuring correct edges is as expected, as the restrictions we define through the background knowledge directly impact this. In general, we can clearly answer *RQ1* in the affirmative.
Tables 3 and 4 show the results for *RQ2*. Generally, we can observe in Table 3 that the percentage of correct edges increases with more data, and the percentage of incorrect edges slightly increases with more data. However, it does not appear that background information is more valuable for more or less data available. As shown in Table 4, the conclusion is a bit more mixed for the methods. The percentage of correct edges for PC actually slightly decreases with more information available; the percentage of incorrect edges decreases quite a bit, though. It should be noted, however, that the ground-truth CPDAGs for PC also vary with information, so the numbers are not directly comparable. HC and LiNGAM greatly benefit from the information. In accordance with previous research, we can observe that HC performs best, whereas PC misses a lot of correct edges and LiNGAM places a lot of wrong edges [8]. For *RQ2*, we can say that background information is generally helpful for all settings but that HC and LiNGAM benefit more from it.

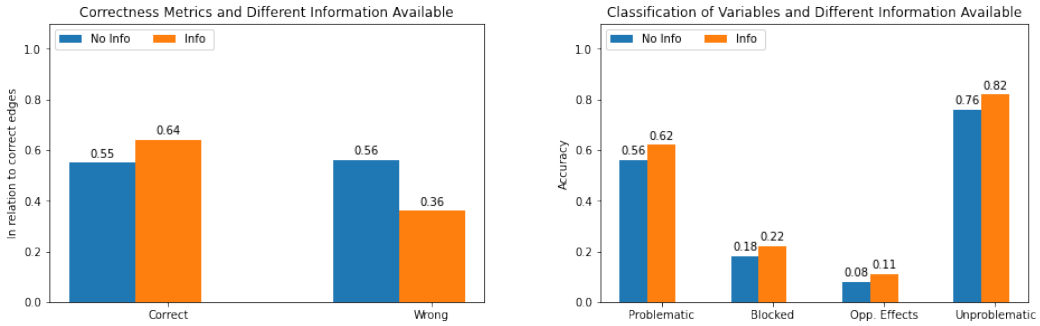
Table 3
Results by sample size and information.

Sample Size	Info	correct edges	wrong edges
500	No Info	0.54	0.54
	Info	0.49	0.33
1000	No Info	0.54	0.55
	Info	0.62	0.35
10000	No Info	0.62	0.59
	Info	0.71	0.41

Table 4
Results by method and information.

Method	Info	correct edges	wrong edges
PC	No Info	0.54	0.33
	Info	0.55	0.22
HC	No Info	0.58	0.38
	Info	0.73	0.19
LiNGAM	No Info	0.52	0.98
	Info	0.69	0.68

Figure 3 (b) answers *RQ3*. We can see that most of the problematic and unproblematic variables



(a) (b)

Figure 3: The average correctness of the learned DAGs (a) and the average percentage of correctly classified variables (b).

are found. This positive result is not true for the other two classes. However, looking deeper into the data, HC actually finds roughly 67% of all blocked variables when having information available, whereas PC and LiNGAM do much worse and push down the average. HC also finds more than 75% of all problematic variables when having information available; LiNGAM is performing above average, too. Finding situations that are indicative of opposing effects variables is tough for all methods, although HC still does much better than average (roughly 35% when having information available). Looking deeper into our results, we observed that the variables are often misclassified as problematic. In this way, at least, attention is drawn to potential problems with them. Most importantly, Figure 3 (b) shows that providing information helps classify sensitive variables, confirming *RQ3*. The difference is not very large, though.

In general, using the background information helps with learning more correct DAGs and classifying the sensitive variables. HC and LiNGAM greatly benefit from background information, and PC-Stable does so less. The above evaluation only serves to show that background information improves the correctness of DAGs. Because of space constraints, we did not add an evaluation of how well the problematic structures are detected (i.e., the paths along which problematic influences might exist). This evaluation will be added in future work. The correctness of the DAGs can be further improved by focusing on score-based methods and potentially adding even more background information (e.g., some sensitive variables may be influenced by other sensitive variables). We want to highlight that *causalfair* allows us to specify more background knowledge, such as whether specific variables must be connected.

6. Conclusion and Limitations

6.1. Limitations

As a general limitation, we want to highlight that the learned DAGs may not reflect real causal relationships. Either important predictive variables (e.g., as highlighted by the discussion of structure (3b) in section 3) or sensitive variables that have an effect might simply not be in the data. While this is a general problem of measuring fairness, it is important to stress that our

CBNs do not necessarily provide a complete picture of the causal mechanisms producing the target.

Moreover, structure learning algorithms have limitations when the data is extremely imbalanced, contains many missing values, and when the relationship between variables is non-linear and complex. In other words, real data could pose a challenge. Real data has rarely been used to evaluate structure learning algorithms, in general, [8], but doing so is, of course, very important. Thus, future research should focus on a real-life evaluation as well.

6.2. Conclusion and Outlook

In this paper, we introduced a classification of sensitive variables into four classes depending on whether and how they are involved in causal structures that could be problematic in the ML context. Additionally, we showed that we can improve the data-driven learning of DAGs by using background knowledge we naturally have in fairness settings. These contributions are implemented in our Python package *causalfair*. We hope researchers and practitioners use this package to evaluate whether they have problematic relationships in their data before learning ML models. In the future, we plan to add more structure learning methods (particularly score-based) to the package. Furthermore, we believe that future research should focus on performing more targeted bias mitigation that can also handle it if we only consider some but not all paths from a sensitive variable to a target as problematic. Chiappa and Isaac discuss a technique to estimate the path-specific effects of variables [5], and we believe this is a good starting point. Moreover, we believe that more effort should be put into constructing accurate DAGs. We show in this paper that background knowledge helps immensely in learning better DAGs, and we believe that further advancing the learning of DAGs using background knowledge should be a future research endeavor.

Acknowledgments

Lea Cohausz is funded by the grant "Consequences of Artificial Intelligence for Urban Societies (CAIUS)," by Volkswagen Foundation.

References

[1] D. Ueda, T. Kakinuma, S. Fujita, K. Kamagata, Y. Fushimi, R. Ito, Y. Matsui, T. Nozaki, T. Nakaura, N. Fujima, et al., Fairness of artificial intelligence in healthcare: review and recommendations, Japanese Journal of Radiology 42 (2024) 3–15.

[2] R. S. Baker, A. Hawn, Algorithmic bias in education, International Journal of Artificial Intelligence in Education (2021) 1–41.

[3] P. Birzhandi, Y.-S. Cho, Application of fairness to healthcare, organizational justice, and finance: a survey, Expert Systems with Applications 216 (2023) 119465.

[4] A. Castelnovo, R. Crupi, G. Greco, D. Regoli, I. G. Penco, A. C. Cosentini, A clarification of the nuances in the fairness metrics landscape, Scientific Reports 12 (2022) 4209. URL: https://www.nature.com/articles/s41598-022-07939-1. doi:10.1038/s41598-022-07939-1.

[5] S. Chiappa, W. S. Isaac, A causal bayesian networks viewpoint on fairness, Privacy and Identity Management. Fairness, Accountability, and Transparency in the Age of Big Data: 13th IFIP WG 9.2, 9.6/11.7, 11.6/SIG 9.2. 2 International Summer School, Vienna, Austria, August 20-24, 2018, Revised Selected Papers 13 (2019) 3–20.
[6] C. Ashurst, R. Carey, S. Chiappa, T. Everitt, Why fair labels can yield unfair predictions: Graphical conditions for introduced unfairness, in: Proceedings of the AAAI Conference on Artificial Intelligence, volume 36, 2022, pp. 9494–9503.
[7] L. Cohausz, J. Kappenberger, H. Stuckenschmidt, What fairness metrics can really tell you: A case study in the educational domain (2024).
[8] M. Scutari, C. E. Graafland, J. M. Gutiérrez, Who learns better bayesian network structures: Accuracy and speed of structure learning algorithms, International Journal of Approximate Reasoning 115 (2019) 235–253.
[9] M. Scanagatta, A. Salmerón, F. Stella, A survey on bayesian network structure learning from data, Progress in Artificial Intelligence 8 (2019) 425–439.
[10] J. Pearl, Causality, Cambridge university press, 2009.
[11] R. S. Baker, L. Esbenshade, J. Vitale, S. Karumbaiah, et al., Using demographic data as predictor variables: a questionable choice, Journal of Educational Data Mining 15 (2023) 22–52.
[12] L. Cohausz, A. Tschalzev, C. Bartelt, H. Stuckenschmidt, Investigating the importance of demographic features for edm-predictions (2023).
[13] L. Grinsztajn, E. Oyallon, G. Varoquaux, Why do tree-based models still outperform deep learning on typical tabular data?, Advances in neural information processing systems 35 (2022) 507–520.
[14] B. Hicks, K. Kitto, L. Payne, S. Buckingham Shum, Thinking with causal models: A visual formalism for collaboratively crafting assumptions, in: LAK22: 12th International Learning Analytics and Knowledge Conference, 2022, pp. 250–259.
[15] N. K. Kitson, A. C. Constantinou, Z. Guo, Y. Liu, K. Chobtham, A survey of bayesian network structure learning, Artificial Intelligence Review (2023) 1–94.
[16] A. C. Constantinou, Z. Guo, N. K. Kitson, The impact of prior knowledge on causal structure learning, Knowledge and Information Systems (2023) 1–50.
[17] S. Shimizu, Lingam: Non-gaussian methods for estimating causal structures, Behaviormetrika 41 (2014) 65–98.
[18] M. Scutari, M. M. Scutari, H.-P. MMPC, Package 'bnlearn', Bayesian network structure learning, parameter learning and inference, R package version 4 (2019).