DAGnosis: Localized Identification of Data Inconsistencies using Structures

Nicolas Huynh* University of Cambridge

Jeroen Berrevoets* University of Cambridge

Nabeel Seedat University of Cambridge

Jonathan Crabbé University of Cambridge

Zhaozhi Qian University of Cambridge

Mihaela van der Schaar University of Cambridge

Abstract

Identification and appropriate handling of inconsistencies in data at deployment time is crucial to reliably use machine learning models. While recent data-centric methods are able to identify such inconsistencies with respect to the training set, they suffer from two key limitations: (1) suboptimality in settings where features exhibit statistical independencies, due to their usage of compressive representations and (2) lack of localization to pinpoint why a sample might be flagged as inconsistent, which is important to guide future data collection. We solve these two fundamental limitations using directed acyclic graphs (DAGs) to encode the training set's features probability distribution and independencies as a structure. Our method, called DAGnosis, leverages these structural interactions to bring valuable and insightful data-centric conclusions. DAGnosis unlocks the localization of the causes of inconsistencies on a DAG, an aspect overlooked by previous approaches. Moreover, we show empirically that leveraging these interactions (1) leads to more accurate conclusions in detecting inconsistencies, as well as (2) provides more detailed insights into why some samples are flagged.

Proceedings of the 27th International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

1 INTRODUCTION

No Data, No Machine Learning. Data plays a crucial role in machine learning as it is used to train and test models (Park et al., 2021; Jain et al., 2020; Sambasivan et al., 2021). To ensure reliable downstream performance, it is essential to have structured mechanisms to assess new data in relation to our training data (Seedat et al., 2022a; Saria and Subbaswamy, 2019). This is a critical concern which should be addressed as neglecting it may lead to poor downstream performance for models evaluated on incongruous samples (Polyzotis et al., 2017; Renggli et al., 2021). This consideration motivates recent interest in *data-centric AI* (DCAI) (Liang et al., 2022; Seedat et al., 2022b), which aims to develop "systematic methods to evaluate [...], the data used to train and test the AI model" (Liang et al., 2022). Building such data-centric methods confers the immediate advantage of flexibility, as insights about the data can be applied to any downstream model.

Inconsistencies. A key challenge in DCAI is to flag inconsistencies in new data with respect to the training set. Inconsistencies can manifest in real-world settings for a variety of reasons. Even if the new samples are in-distribution, they may exhibit inconsistencies due to finite-sample effects exacerbated by regions of low data coverage (e.g. underrepresented subgroups) (Krawczyk, 2016; Yuksekgonul et al., 2023), or data biases (Torralba and Efros, 2011) in the training dataset. The identification of these inconsistencies is of paramount importance to ensure reliable downstream performance and can guide future data collection. It justifies a systematic and principled data-centric approach, leading to *rich and valuable insights*.

Tabular Data and Sparse Connections. Of particular interest in this paper is tabular data, which is ubiquitous in real-world and high-stake settings, such as medicine, finance or economics (Borisov et al., 2021).

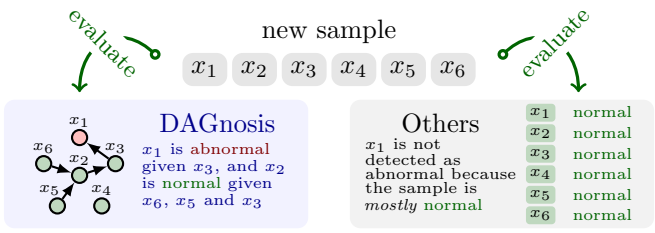


Figure 1: **DAGnosis Provides *Precise* Analysis.** DAGnosis takes a radically different approach compared to other data-evaluation methods. Rather than evaluating each dimension of a new sample in relation to all the other dimensions, we evaluate in relation to *the structure* of the sample. This may lead to different samples being flagged while giving interpretation for that conclusion.

Data-SUITE (Seedat et al., 2022a) is the method most relevant to our work, as it flags inconsistencies in the tabular domain. It computes feature-wise uncertainty, in the form of prediction intervals. One key element of the approach is the input to obtain these intervals, which is a compressive representation (e.g. PCA or autoencoder) of the *complete* input. However, using such compressive representations is suboptimal for two reasons. First, it overlooks the sparse dependencies between features in tabular datasets (Yang et al., 2022; Kalisch and Bühlman, 2007) (which differ from images or text where features are very tightly coupled). In such datasets, not all the features might be relevant (Jordon et al., 2018). Second, it does not permit the *localization* of the reasons why a sample is deemed inconsistent, which is problematic from an auditing perspective. We discuss this in more details in Section 2.

To address these limitations, we present DAGnosis, a data-centric evaluation strategy for the tabular domain. *DAGnosis addresses the problem of flagging inconsistencies,* for which it requires two components: a way to leverage sparsity and independence in the tabular data, where we leverage structures modeled as *directed acyclic graphs* (DAGs); and a way to flag instances in the data, where we build on *conformal prediction*. DAGnosis provides *localized* instance-wise conclusions: having flagged an inconsistency, it gives a set of features which explain it. Localization is important because it can inform future data collection (enriching the *training set*) or suggest new measurements of features (correcting the *test samples* which exhibit measurement noise). In Figure 1 we see that DAGnosis crucially relies on a Bayesian network describing the features, which makes it *interpretable by design* (Barredo Arrieta et al., 2020). Unlike previous data-centric methods, our conclusions take into account the interactions between features through the structure encoded in a DAG.

Contributions. DAGnosis advances the state-of-the-art as follows: ① **Conceptually**, DAGnosis identifies and addresses the suboptimality of compressive representations. To the best of our knowledge, DAGnosis is the first method to leverage structures for data-centric insights. It unlocks the localization of the reasons why a sample is deemed inconsistent, an aspect overlooked by the state-of-the-art. ② **Technically**, DAGnosis learns a DAG describing the relationships between the features and trains feature-wise conformal predictors. It conditions them on relevant variables as determined by the DAG. ③ **Empirically**, we demonstrate in Section 4 that DAGnosis outperforms the SOTA on accuracy of inconsistency detection and downstream accuracy when deferring predictions on inconsistent samples. Furthermore, we provide a detailed case study on a real-world dataset in Section 5, showing how practitioners can benefit from DAGnosis to gain understanding of inconsistencies.

2 RELATED WORK

Data-centric Evaluation. Even though data-centric insights are important, they have been mostly neglected in favor of model-dependent conclusions. This is epitomized by the field of predictive uncertainty quantification (Gawlikowski et al., 2021), where the idea is to categorize samples with respect to the uncertainty in the prediction of a given model (e.g. with Gaussian Processes (Rasmussen and Williams, 2003), Bayesian Neural Networks (Ghosh et al., 2018) or using ensembles (Lakshminarayanan et al., 2016)). In this work, we move away from this branch and instead give conclusions with respect to the data itself.

As such, Data-SUITE (Seedat et al., 2022a) is the method most relevant to our work. However, it has several key limitations:

(i) non-adaptiveness of the conditioning sets. The same input (i.e. the representation obtained with PCA) is given to d feature regressors, while in practice each feature might depend on a different set of variables. DAGnosis creates a set of conditioning variables specific to each feature. In this way, it is more flexible with respect to the specificity of each feature.

(ii) localization. Data-SUITE does not offer a localized explanation on why examples are flagged as inconsistent in terms of the input features them-

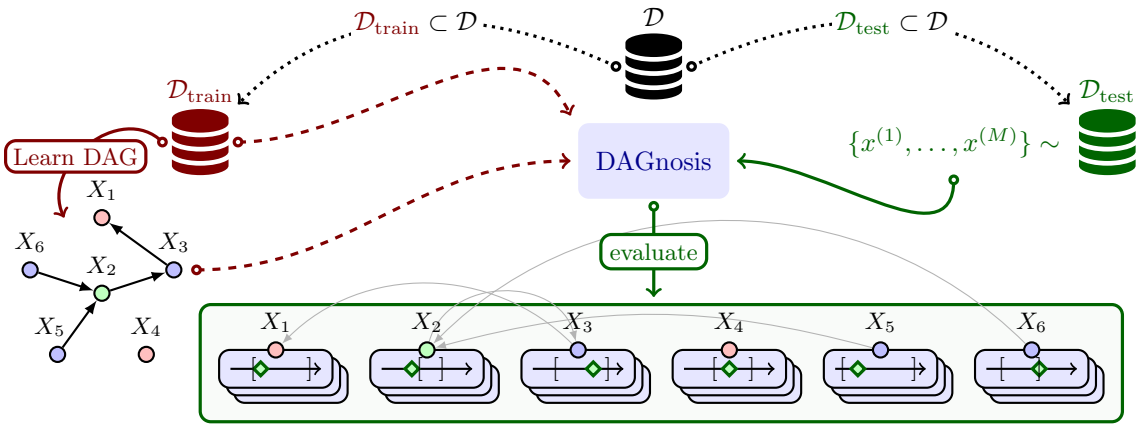


Figure 2: **High-level Overview of DAGnosis.** DAGnosis evaluates samples in a test-bed dataset $\mathcal{D}_{\text{test}}$. It first learns a DAG (using a variety of structure learners). Next, DAGnosis builds prediction intervals for every feature using conformal prediction. They are conditioned on smart subsets of the data's features, informed by the DAG.

selves; because the features are combined in the compressive representation, it is difficult to contextualise an inconsistent feature as resulting from an abnormal value conditioned on the other features. DAGnosis unlocks localization, and provides important insights which can guide future data collection.

(iii) sparse interactions between the features. Sparse interactions are ubiquitous in real settings, evidenced by abundant noise variables (Kalisch and Bühlman, 2007). In such settings, using a compressive representation leads to a loss of information, affecting the quality of the conformal predictors, and hence the data-centric conclusions.

Structure. In order to account for the sparse interactions between features in the tabular domain — an aspect overlooked by Data-SUITE — DAGnosis leverages structures (DAGs). A DAG consists of vertices and edges, where the vertices represent the random variables comprising a feature set, and the edges model direct dependence (Koller and Friedman, 2009; Guyon et al., 2007; Berrevoets et al., 2023a). We term the setting *sparse* (Ng et al., 2020) when the number of edges is low. It is these sparse settings that we hope to model more accurately with DAGnosis.

DAGs can be discovered, via a variety of structure learners (Zheng et al., 2018; Peters et al., 2017; Verma and Pearl, 1990b,a; Geiger and Heckerman, 1994; Berrevoets et al., 2023b), or instead be provided or completed by the user, when prior knowledge is available (Hasan and Gani, 2022; Sinha and Ramsey, 2021; Sachs et al., 2005).

3 DAGNOSIS: IDENTIFYING INCONSISTENCIES USING STRUCTURES

We are interested in scenarios where we have a training dataset and want to flag inconsistent test samples without relying on a downstream model. Moreover, we wish to go beyond the current data-centric capabilities of *just* flagging samples. We also want to provide a *reason or localization* as to why they were flagged. This is important to make data-centric methods principled from an auditing perspective.

Data. We consider a d-dimensional feature space $\mathcal{X} \subseteq \mathbb{R}^d$, where we have access to a training dataset, $\mathcal{D}_{\text{train}} = \{x^k \mid k \in [n_{\text{train}}]\} \subset \mathcal{X}$, and a test set $\mathcal{D}_{\text{test}} = \{x^j \mid j \in [n_{\text{test}}]\} \subset \mathcal{X}$, with n_{train} and n_{test} being the cardinalities of $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$, respectively. Moreover, we assume $\mathcal{D}_{\text{train}}$ is composed of i.i.d. samples coming from a distribution P^*. We denote indices of features with subscripts, i.e. x_i is the i-th feature of x. When $\mathcal{S}$ is a set of indices, $x_{\mathcal{S}}$ denotes the restriction of x to the indices in $\mathcal{S}$. For future convenience, we also define $\mathcal{I}(\mathcal{E})$ which returns the feature-indices present in the set of random variables $\mathcal{E}$.

As shown in Figure 2, we wish to flag inconsistencies in $\mathcal{D}_{\text{test}}$ with respect to $\mathcal{D}_{\text{train}}$. In order to characterize inconsistencies in $\mathcal{D}_{\text{test}}$ at deployment time, we aim to provide feature-wise conclusions for every sample $x = [x_1, \ldots, x_d]^\top \in \mathcal{D}_{\text{test}}$, which will be aggregated into sample-wise conclusions. These conclusions inform whether or not a sample is labeled inconsistent.

Structures Representing Data. A key contribution of our work is to approach the problem by leveraging structures. The structures of interest are directed acyclic graphs (DAG). Intuitively these structures act

as a compact representation of a factorization of P^* (Koller and Friedman, 2009, Chapter 3 & 4).

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a DAG, comprised of a set of vertices ($\mathcal{V}$) and edges ($\mathcal{E}$), with $\mathcal{E} \subset \mathcal{V} \times \mathcal{V}$. If $(V_i, V_j) \in \mathcal{E}$ then $(V_j, V_i) \notin \mathcal{E}$, making $(V_i, V_j) \neq (V_j, V_i)$. In practice, we consider Bayesian networks (BN) describing $\mathbf{X}$, a random vector following the training distribution P^*, i.e. $\mathbf{X} = (X_V)_{V \in \mathcal{V}} \in \mathcal{X}$ is a random vector with its coordinates in correspondence with the graph's vertices. The whole point of using Bayesian networks is that they encode (conditional) independencies between the features graphically, with the notion of d-separation (Geiger et al., 2013). As a direct consequence, for each feature X_i, we can identify a minimal set of features to best regress X_i, according to the BN. We first recall the definition of Markov blankets.

Definition 3.1 (Markov blanket)**.** Let $\mathcal{S} = \{X_1, X_2, ..., X_d\}$ be a set of random variables. For $i \in [d]$, a Markov blanket of the random variable X_i in $\mathcal{S}$ is any subset $\mathcal{S}_i \subseteq \mathcal{S}$ such that:

$$X_i \perp\!\!\!\perp (\mathcal{S} \setminus \mathcal{S}_i) \mid \mathcal{S}_i$$

A minimal Markov blanket is called a Markov boundary.

Definition 3.2 (Markov boundary)**.** A Markov boundary of a random variable X_i in a set $\mathcal{S} \coloneqq \{X_1, \ldots, X_d\}$ is any subset $\mathcal{S}^- \subset \mathcal{S}$ which is a Markov blanket (Definition 3.1), but does not contain any proper subset which itself is a Markov blanket of X_i. We will denote the Markov boundary of X_i as $\mathcal{X}^-(X_i)$.

A Markov boundary enables us to find the minimal set of variables which capture all the *sufficient* information to describe a particular feature. Moreover, we can find them graphically in a Bayesian network (Koller and Friedman, 2009). As we will describe in Section 3.1, we use these minimal sets to build conditional conformal predictors. Crucially, this contrasts using an *entire* representation, likely containing irrelevant information, especially in tabular settings with many irrelevant variables (i.e. a sparse setting).

3.1 Structure-based Assessment of Samples

To flag inconsistencies in the data, DAGnosis models feature-wise uncertainty in a frequentist setting using a graphical representation of $\mathcal{D}_{\text{train}}$. Our desideratum is to obtain distribution-free prediction intervals (PIs) for each feature, with a coverage guarantee (in order to control a desired False Positive Rate when flagging inconsistencies). To fulfill this desideratum, DAGnosis leverages conformal prediction (Vovk et al., 2005). Specifically, for all $i \in [d]$ and any $x \in \mathcal{D}_{\text{test}}$ as well a significance level $\alpha \in (0, 1)$, we construct PIs $[l_{i,\alpha}(x), r_{i,\alpha}(x)]$ for x_i. Given α, conformal prediction comes with a marginal coverage guarantee stemming from a calibration step, when the exchangeability assumption is satisfied (Balasubramanian et al., 2014). More details are provided in Appendix A.1.

Rather than directly using the complete x as input to the conformal estimators, we will exploit the structure given by a discovered DAG $\mathcal{G}$— which models conditional dependencies —to come up with more informed PIs in a compact and accurate way. As we will confirm in Section 4, doing so leads to more accurate discovery, as well as improved downstream task performance when deferring prediction on samples flagged as inconsistent.

Structures and Independence. Consider the autoregressive factorization $P(\mathbf{X}) = P(X_1)P(X_2|X_1)...P(X_d|X_1, ..., X_{d-1})$ over d random variables, which holds true for any distribution P. With this factorization and no further assumptions on (conditional) independencies, one can identify the simple Markov boundary, $\mathcal{X}^-(X_i) = \{X_1, ..., X_d\} \setminus \{X_i\}$, which amounts to using all the other variables to describe X_i. However, in many settings, variables exhibit (conditional) independence relationships, yet the approach we have just described (essentially taken by other benchmarks) does not account for it. This insight motivates the use of structures.

We design our method to be agnostic to the way the structure is provided. It can come from a structure learner, which takes as input a dataset $\mathcal{D}$, and outputs a DAG $\mathcal{G}$; or can be given a priori. As such, we can choose any structure learner in the wide range of conditional independence testing based (CIT), like the PC algorithm (Spirtes et al., 2000), or score-based methods. In some settings (Sachs et al., 2005; Mooij et al., 2016; Pinna et al., 2013), we can leverage prior knowledge and provide (or complete) the underlying ground-truth DAG. In our experiments, we assume no access to prior knowledge and thus learn the DAG ourselves, in light of fair comparison to other benchmark methods.

Constructing Feature-wise Prediction Intervals. Given a DAG $\mathcal{G}$ as input, we compute adaptive prediction intervals for each feature X_i, and any sample $x \in \mathcal{X}$, denoted by $[l_{i,\alpha}(x), r_{i,\alpha}(x)]$, where α is the significance level. We use Conformalized Quantile Regression (CQR) Romano et al. (2019) as our inductive conformal prediction method, which has been shown to outperform other inductive conformal prediction benchmarks. To perform conformal prediction, we split $\mathcal{D}_{\text{train}}$ into a training set $\mathcal{D}^+_{\text{train}}$ and a calibration set $\mathcal{D}_{\text{cal}}$. We now describe the construction of the feature-wise prediction intervals. The following steps are conducted for each feature $i \in [d]$:

◆ **Step 1** Given two significance levels α_{lo}, α_{hi}, train conditional quantile regressors $\hat{q}_{i,\alpha_{lo}}, \hat{q}_{i,\alpha_{hi}}$, using $\mathcal{D}^+_{\text{train}}$.

Algorithm 1 – Training. We describe how we train each feature-specific conformal predictor, informed by the DAG. While we consider $\mathcal{G}$ as an input to the training routing, $\mathcal{G}$ can also be learned prior to this.

Input: A proper training set $\mathcal{D}^{+}_{\text{train}}$, a calibration set $\mathcal{D}_{\text{cal}}$, significance levels α, α_{lo} , α_{hi}, and a DAG $\mathcal{G}$
Output: A list of conformal predictors $\{(l_{i,\alpha}, r_{i,\alpha}) : i \in [d]\}$
for $i \in [d]$ **do**
  $\mathcal{D}_i \leftarrow \{(f_i(x), x_i) : \forall x \in \mathcal{D}^{+}_{\text{train}}, \forall i \in [d]\}$ {Construct the training set using $\mathcal{G}$}
  $\hat{q}_{i,\alpha_{lo}}$`.train(`α_{lo}`,`$\mathcal{D}_i$`)` and $\hat{q}_{i,\alpha_{hi}}$`.train(`α_{hi}`,`$\mathcal{D}_i$`)` {Fit the quantile regressors}
  $\epsilon_{i,\alpha} \leftarrow$`calibrate(`$\hat{q}_{i,\alpha_{lo}}, \hat{q}_{i,\alpha_{hi}}, \mathcal{D}_{\text{cal}}$`)` {Calibrate the quantile regressors}
  $l_{i,\alpha} \leftarrow \hat{q}_{i,\alpha_{lo}} - \epsilon_{\alpha,i}$ and $r_{i,\alpha} \leftarrow \hat{q}_{i,\alpha_{hi}} + \epsilon_{\alpha,i}$ {Define the conformal predictors}
  Add $l_{i,\alpha}$ and $r_{i,\alpha}$ to the list of conformal predictors
end for

The input to the quantile regressors for feature i is denoted by $f_i(x)$. The quantile regressors are trained using the α_{lo} and α_{hi}-pinball losses.

Crucial to DAGnosis is our (graph-based) definition of f_i, which is $f_i(x) \coloneqq x_{\mathcal{I}(\mathcal{X}^{-}(X_i))}$. This definition directly leverages the structure $\mathcal{G}$ to define f_i. Alternatively stated, $f_i(x)$ contains all the feature-indices present in X_i's Markov boundary. This is in stark contrast to methods which use a compressing representation (PCA/Autoencoder) where x_i could indirectly be present.

◆ **Step 2** Use a calibration set $\mathcal{D}_{\text{cal}}$ to compute a set of non-conformity scores $C_i = \{E_{i,j} \mid j \in [|\mathcal{D}_{\text{cal}}|]\}$, and compute the $(1-\alpha)(1+\frac{1}{|\mathcal{D}_{\text{cal}}|})$ empirical quantile of C_i, denoted as $\epsilon_{\alpha,i}$ (see Appendix A.1 for details). The prediction intervals for a particular sample x and a feature $i \in [d]$ are then given by $[\hat{q}_{i,\alpha_{lo}}(f_i(x)) - \epsilon_{\alpha,i}, \hat{q}_{i,\alpha_{hi}}(f_i(x)) + \epsilon_{\alpha,i}]$. We summarize the training part of DAGnosis in Algorithm 1.

Remark: DAGnosis does not require learning a full probabilistic model of the data. It solely needs a structure before performing feature-wise CP. CP is a distribution-free paradigm, hence DAGnosis makes no specific distributional assumption.

3.2 Leveraging Structure to Flag Inconsistencies

In the previous section we detailed how DAGnosis builds feature-wise conformal predictors. We can now use these predictors to flag inconsistencies in new test examples with respect to the training set.

Benefits of Using a Structure. Using a structure in DAGnosis directly solves problems related to other data-centric methods when attempting to flag inconsistent examples. Indeed, (1) for every feature, the structure defines which set of conditioning variables we should use; (2) by virtue of having an explicit set of conditioning variables, we localize the causes of the inconsistencies; (3) we account for irrelevant variables, which are isolated nodes in the structure, and which are not used as input to the conformal prediction step.

Metrics. Similar to Seedat et al. (2022a), the conformal prediction framework brings an interesting conclusion at the sample-level, which is whether or not each feature value falls inside its associated confidence interval. This is encapsulated in the notion of inconsistency.

Definition 3.3 (Inconsistency)**.** Let $x \in \mathcal{D}_{\text{test}}$ be a test instance for which we construct a $(1-\alpha)$-PI, $[l_{i,\alpha}(x), r_{i,\alpha}(x)]$, for each feature $x_i, i \in [d]$, with a significance level $\alpha \in (0,1)$. For each $x_i, i \in [d]$, the *feature inconsistency* is a binary variable indicating if x_i falls out of the PI.

$$\nu_i(x) \equiv \mathbb{I}(x_i \notin [l_{i,\alpha}(x), r_{i,\alpha}(x)]) \tag{1}$$

The *instance inconsistency* $\nu(x)$ is obtained by summing the feature inconsistencies $\nu_i(x)$, that is $\nu(x) = \sum_{i=1}^{d} \nu_i(x)$. A sample x is inconsistent if $\nu(x) > 0$.

With this definition, we can now give conclusions on the data *itself*, summarized in Algorithm 2 in Appendix A.2.

Remark: We compare DAGnosis to conventional out-of-distribution detectors in Appendix A.4, highlighting that DAGnosis can flag in-distribution inconsistencies like Data-SUITE (Seedat et al., 2022a).

4 EXPERIMENTS

In this section, we show that taking into account structures leads to *accurate* and *localized* detection of inconsistencies. We also demonstrate how DAGnosis enables reliable downstream performance. For details on the experimental setup [1] [2], please refer to Appendix B.

Baselines. For the rest of this section, we consider DAGnosis and the baseline Data-SUITE as it is the only comparable method. We consider different variants of

[1] https://github.com/nicolashuynh/DAGNOSIS
[2] https://github.com/vanderschaarlab/DAGNOSIS

DAGnosis, where the nature of the DAG used by DAGnosis differs among the variants: *GT* (ground-truth DAG describing P^*, when it is known), *Autoregressive* (autoregressive factorization, described in Section 3.1), *NOTEARS* (NT) (Zheng et al. (2020)) and *DAGMA* (Bello et al. (2022)), two differentiable structure learners, and *PC* (Spirtes et al. (2000)), a constraint-based structure learner. For Data-SUITE, we use $\frac{d}{2}$ as the representation dimension similarly to Seedat et al. (2022a), and use CQR as the Inductive Conformal Prediction (ICP) method, since it is considered the SOTA for ICP.

4.1 DAGnosis Flags Inconsistencies Accurately

Dataset. For this first experiment, we generate synthetic data in three steps: *Step 1)* Sample a DAG $\mathcal{G}$, using an Erdős–Rényi model with parameters (d, s), where d is the feature space dimension, and s is the number of edges, which controls the sparsity of $\mathcal{G}$. *Step 2)* Sample Structural Equation Models (SEM), either linear or two-layered multilayer-perceptrons (MLP). *Step 3)* Sample the data using a topological ordering induced by $\mathcal{G}$. We consider an additive noise model, with Gaussian noise of mean 0 and variance 1. The parameters of the SEMs are sampled from a mixture of two uniform distributions: $\mathcal{U}(-2.5, -0.5)$ and $\mathcal{U}(0.5, 2.5)$, following Zheng et al. (2018).

We then corrupt the sampled SEMs to create inconsistencies at test-time: we add Gaussian noise to the parameters of the linear SEMs and when the SEMs are MLPs, we perform the corruption on the last layer, by sampling 5 dimensions to be corrupted. More details are included in Appendix B.

Methodology. The goal of this experiment is to show that structures permit to flag inconsistencies more accurately. To show this, the methods are evaluated on their ability to flag inconsistent samples in a test set $\mathcal{D}_{\text{test}}$. We consider $d = 20$, $n_{\text{train}} = 1000, s \in \{10k \mid k \in [4]\}$ (controlling the sparsity in the structure), and set the same significance level $\alpha = \frac{0.1}{d}$ for every feature and every method, thereby adopting the Bonferroni correction. For every s, we sample 20 DAGs, SEMs and corresponding training sets. At test time, we sample and corrupt 2 features, by altering their corresponding SEMs. We then sample $\mathcal{D}_{\text{test,corrupt}}$, with $n_{\text{test}} = 10000$. In order to investigate false positives, we also sample a clean test set drawn from the same distribution as the training dataset, denoted as $\mathcal{D}_{\text{test,clean}}$, with cardinality n_{test}. The final test set is $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{test,corrupt}} \bigcup \mathcal{D}_{\text{test,clean}}$.

Results. The detection task involves two classes (1: corrupted, 0: non-corrupted). Hence, for each s, we report the F_1 scores, precision, and recall for the different methods. As seen in Table 1, incorporating structures is consistently useful, but it is especially useful in sparse settings ($s \in \{10, 20\}$): DAGnosis does not take into account noise variables thanks to the structure. As a direct consequence, this leads to better detection results than DAGnosis Auto, which uses $d-1$ variables for each feature, and Data-SUITE CQR, which uses PCA as a representation. Beyond these metrics, we also compute an AUROC for each s and each method, sweeping α across $\{\frac{0.1k}{d} \mid k \in [9]\}$. We report the results in Appendix C.2 and show that DAGnosis again outperforms Data-SUITE.

Takeaway. Structures permit to model and take into account feature interactions. Incorporating the conditional independencies embodied in these structures makes it possible to specialize the sets of conditioning variables and ignore irrelevant variables. This leads to a better detection of inconsistencies compared to other representations of $\mathcal{D}_{\text{train}}$ which ignore this information.

4.2 DAGnosis is Effective Even With Imperfect DAGs

Methodology. In this experiment, we investigate the ability of DAGnosis to flag inconsistencies using imperfect DAGs. For that, we consider a high-dimensional synthetic setting ($d = 100$) where we either corrupt the ground-truth DAG, or learn it with a structure learner. We generate $k = 5$ DAGs, with $s = 50$. We consider a list of Structural Hamming Distances (SHD), namely [10, 20, 30, 40]. For each of these SHD values, we sample 5 corrupted DAGs with the given SHD from the ground-truth DAG. This mechanism directly mimics misspecifications of the DAG of various strengths, where a high SHD indicates a high misspecification. We also learn the DAG with the structure learner DAGMA (Bello et al., 2022), illustrating the flexibility of DAGnosis in the way the structure is learnt. Finally, we sample the data with MLP SEMs, as in Section 4.1.

Results. F1 score, precision and recall for the inconsistency detection task are reported in Table 2. For each SHD, we average the results over the k misspecified DAGs which we sampled. These results show that DAGnosis outperforms Data-SUITE and achieves good performance even when the input DAG does not match exactly the ground-truth DAG, highlighting its robustness. We report additional results in high-dimension ($d = 200$) in Appendix C.3.

Takeaway. DAGnosis is robust to misspecifications of the DAG and can operate in high-dimensional setups.

4.3 DAGnosis Unlocks Localization

Methodology. Having demonstrated that DAGnosis flags inconsistencies accurately in Section 4.1, we now

Table 1: **Results on Inconsistency Detection.** We report the F_1 score (↑), precision (prec.) (↑) and recall (rec.) (↑) for the inconsistency detection task over different settings with decreasing sparsity (higher s indicates less sparse). Mean and 1.96× standard errors are reported. We benchmark against DAGnosis paired with a naive DAG modeling the autoregressive factorization (***DN Auto***), Data-SUITE (***DS CQR***), and have colored our method's (***DN NT***) row for clarity. Beyond the relevant benchmarks, we have also included DAGnosis with a ground truth DAG ***(DN GT)***. This acts as an oracle "upper bound" and is shaded to clearly distinguish it from the other methods. Our results show that DAGnosis (using NOTEARS) improves F_1 scores, precision, and recall.

	$s = 10$			$s = 20$			$s = 30$			$s = 40$		
	F_1 (↑)	Prec. (↑)	Rec. (↑)	F_1 (↑)	Prec. (↑)	Rec. (↑)	F_1 (↑)	Prec. (↑)	Rec. (↑)	F_1 (↑)	Prec. (↑)	Rec. (↑)
	Linear SEMs											
DN Auto	0.81 (.03)	0.89 (.01)	0.76 (.04)	0.84 (.03)	0.89 (.01)	0.79 (.04)	0.86 (.03)	**0.91** (.01)	0.82 (.04)	0.86 (.02)	0.91 (.01)	0.83 (.04)
DS CQR	0.81 (.03)	0.88 (.01)	0.75 (.04)	0.84 (.03)	0.90 (.01)	0.80 (.04)	0.85 (.03)	0.90 (.01)	0.81 (.04)	0.85 (.02)	0.91 (.01)	0.80 (.04)
DN NT	**0.85** (.02)	**0.89** (.01)	**0.82** (.04)	**0.88** (.02)	**0.90** (.01)	**0.86** (.03)	**0.87** (.03)	0.90 (.01)	**0.85** (.04)	**0.88** (.02)	**0.91** (.01)	**0.85** (.04)
(DN GT)	0.85 (.02)	0.90 (.01)	0.82 (.04)	0.88 (.02)	0.90 (.01)	0.86 (.03)	0.87 (.02)	0.90 (.01)	0.85 (.04)	0.88 (.02)	0.91 (.01)	0.85 (.04)
	MLP SEMs											
DN Auto	0.78 (.07)	0.89 (.02)	0.72 (.09)	0.83 (.08)	0.89 (.03)	0.81 (.11)	0.79 (.06)	0.88 (.02)	0.74 (.09)	0.76 (.1)	0.86 (.05)	0.73 (.12)
DS CQR	0.75 (.07)	0.88 (.02)	0.67 (.09)	0.79 (.1)	0.87 (.04)	0.77 (.12)	0.73 (.09)	0.86 (.03)	0.67 (.11)	0.76 (.11)	0.85 (.05)	0.73 (.14)
DN NT	**0.93** (.02)	**0.91** (.01)	**0.95** (.03)	**0.88** (.06)	**0.90** (.02)	**0.89** (.08)	**0.85** (.05)	**0.89** (.02)	**0.82** (.08)	**0.84** (.06)	**0.89** (.02)	**0.82** (.08)
(DN GT)	0.93 (.01)	0.91 (.01)	0.96 (.02)	0.90 (.04)	0.91 (.01)	0.91 (.06)	0.85 (.05)	0.89 (.01)	0.82 (.08)	0.84 (.07)	0.88 (.02)	0.82 (.09)

Table 2: **Robustness of DAGnosis.** We report the detection metrics of the different methods, for $d = 100$. DAGnosis (DN) is robust to misspecifications of the DAG, when the DAG is either corrupted or learnt with a structure learner such as DAGMA.

Method	F1-score	Precision	Recall
Data-SUITE	0.49 (.23)	0.81 (.07)	0.4 (.27)
DN SHD 10	0.70 (.1)	0.85 (.04)	0.63 (.11)
DN SHD 20	0.73 (.1)	0.86 (.04)	0.67 (.11)
DN SHD 30	0.69 (.1)	0.85 (.04)	0.62 (.11)
DN SHD 40	0.63 (.1)	0.83 (.04)	0.55 (.11)
DN DAGMA	**0.79** (.1)	**0.89** (.03)	**0.72** (.13)

show that structures enable the *localization* of such inconsistencies, which is impossible with Data-SUITE. We consider a synthetic setup, with $d = 4$. The DAG used to generate the data is the chain $X_1 \to X_2 \to X_3 \to X_4$ and the SEMs are MLPs. We only corrupt the last feature in the topological ordering, which is X_4, making the inconsistencies *localized*. Hence we desire to flag inconsistent samples solely on this feature.

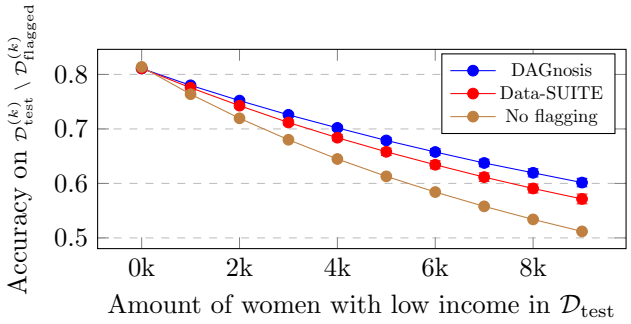
Results. We compute the average number of flagged features for the samples deemed inconsistent by Data-SUITE (DS) and DAGnosis (DN), with $n_{\text{test}} = 10000$, denoted by a_{DS} and a_{DN}. We obtain $a_{\text{DS}} = \mathbf{3.48}$ and $a_{\text{DN}} = \mathbf{1.05}$. DAGnosis is significantly more precise when flagging inconsistent samples, as it most often localizes and flags the inconsistencies only on X_4, while Data-SUITE flags nearly all the features on average.

Takeaway. DAGnosis unlocks localization and flags inconsistencies where they happen. Hence, DAGnosis can answer two questions: is there an inconsistency? If so, *where*? This property stems from its nature, since it uses *conditionals by design*. On the contrary, Data-SUITE can only answer one question: is there an inconsistency?

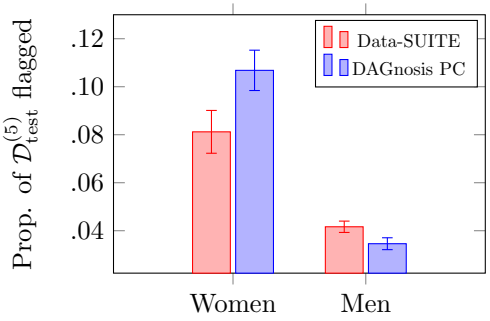
4.4 Reliable Downstream Performance

Methodology. An important use case of DAGnosis is to ensure reliable downstream performance, where a practitioner might want to defer predictions on samples flagged as inconsistent. As an example, we use the dataset **UCI Adult income** (Asuncion, 2007), which captures demographic, financial and personal features, with $d = 14$. We define specific train/test splits to control the presence of inconsistencies in the test dataset. More precisely, we split men equally in $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$. We put women with high incomes in $\mathcal{D}_{\text{train}}$. We gradually add women with low incomes in $\mathcal{D}_{\text{test}}$, with a parameter k controlling the number of such samples, giving a list of k test sets $\mathcal{D}_{\text{test}}^{(k)}$. This motivates identifying inconsistencies in $\mathcal{D}_{\text{test}}^{(k)}$ with respect to $\mathcal{D}_{\text{train}}$, as we expect to flag most of the women in $\mathcal{D}_{\text{test}}^{(k)}$ as inconsistent. We repeat the experiment with 5 different seeds.

Results. We then compute the downstream task accuracy on $\mathcal{D}_{\text{test}}^{(k)} \setminus \mathcal{D}_{\text{flagged}}^{(k)}$, i.e. we defer prediction on the



(a) Downstream accuracy for non-flagged samples



(b) Analysis of flagged samples

Figure 3: **(a):** Deferring prediction on $\mathcal{D}_{\text{flagged}}^{(k)}$, the set of samples flagged by DAGnosis, leads to a better downstream accuracy. **(b):** We report the proportion of test samples which are flagged and are women or men, for both DAGnosis and Data-SUITE (DS). DAGnosis is more accurate than DS because it flags more inconsistent samples, while flagging a similar number of men.

inconsistent samples. As shown in Figure 3a, the presence of inconsistencies has a large impact on the downstream accuracy (mean and 1.96× standard error reported). DAGnosis leads to the highest downstream accuracy on $\mathcal{D}_{\text{test}}^{(k)} \setminus \mathcal{D}_{\text{flagged}}^{(k)}$ (when deferring prediction on inconsistent samples). We emphasize that this result is not a consequence of DAGnosis flagging more men in $\mathcal{D}_{\text{test}}^{(k)}$ than DS. To illustrate this, we report in Figure 3b for $k = 5$ the proportion of $\mathcal{D}_{\text{test}}^{(5)}$ which is flagged and consists of women and men (mean and 1.96× standard error reported). We conclude that both methods flag a similar amount of men. However, the key difference is that DAGnosis flags more women, who are inconsistent by design of the train/test split. We provide additional results for the **Credit** dataset (Yeh and Lien, 2009) in Appendix C.4, similarly showing that DAGnosis enables reliable downstream performance.

Takeaway. DAGnosis informs the practitioner by flagging samples harmful for downstream tasks, for which predictions should be deferred, leading to better downstream performance.

5 HOW TO USE DAGNOSIS STEP-BY-STEP

Having demonstrated DAGnosis' superior accuracy in identifying inconsistencies and enabling reliable downstream performance, we now provide an illustrative walkthrough on how DAGnosis can be valuable to practitioners investigating real-world data. To demonstrate its utility, we present a case study highlighting its ability to *localize* the causes of inconsistencies.

Step 1. Dataset Construction. Throughout this section, we use the real-world dataset **UCI Adult income**. As in Section 4.4, we assume access to $\mathcal{D}_{\text{train}}$ and test sets $\mathcal{D}_{\text{test}}^{(k)}$, $k \in [9]$. For what follows, we take $\mathcal{D}_{\text{test}} = \mathcal{D}_{\text{test}}^{(5)}$. Our aim is to flag samples in $\mathcal{D}_{\text{test}}$ which are inconsistent with respect to $\mathcal{D}_{\text{train}}$.

Step 2. DAG Discovery. Since no ground-truth DAG is available for this dataset, we use the PC algorithm and discover a DAG $\mathcal{G}$, prior to using DAGnosis.

Step 3. Flagging Inconsistencies. Equipped with $\mathcal{G}$, we perform the machinery of DAGnosis detailed in Section 3.1, by training the conformal predictors using $\mathcal{D}_{\text{train}}$ and $\mathcal{G}$. We then obtain the set $\mathcal{D}_{\text{flagged}}$, i.e. the samples in $\mathcal{D}_{\text{test}}$ which are flagged as inconsistent.

Having identified a set of inconsistent samples $\mathcal{D}_{\text{flagged}}$, a practitioner may desire to understand these inconsistencies with respect to $\mathcal{D}_{\text{train}}$. DAGnosis empowers the practitioner in this regard because it brings *localization*.

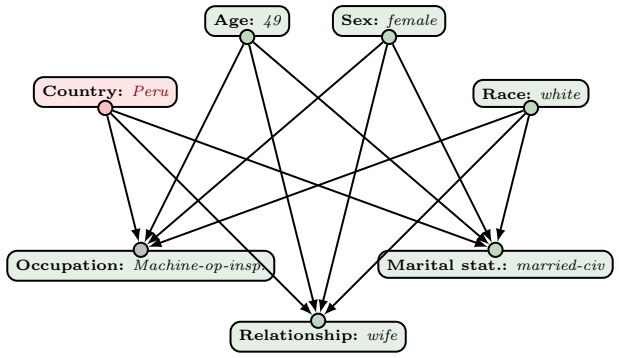


Figure 4: We depict the Markov boundary for the feature *Country*, which is flagged for the given example. An investigation of $\mathcal{D}_{\text{train}}$ shows that this inconsistency can be traced back to the *Occupation* feature.

Step 4. Localizing the Inconsistencies. Figure 4 presents an example of a sample flagged by DAGnosis (and ignored by Data-SUITE). This sample is also wrongly predicted by the downstream classifier, hinting at its inconsistency. It is flagged by DAGnosis on the feature *Native-country* (Peru). We show in Figure 4 the associated Markov boundary as well as the accompanying feature values. This MB is of size 6, less

than half of the total number of features (14). Hence DAGnosis informs the practitioner with a narrow set of variables which explain the inconsistency.

Step 5. Gaining Understanding with $\mathcal{D}_{\text{train}}$. One feature of particular interest in this MB is *Occupation*. We can then go back to $\mathcal{D}_{\text{train}}$ to understand the inconsistency, narrowing down our focus to the relationship between *Occupation* and *Country*. We notice that all the women who exhibit *Occupation = Machine-op-insp.* in $\mathcal{D}_{\text{train}}$ are from the *United States* (which is to be expected since women in the training set have an income $>$ 50k by construction). This conflicts with the the value *Peru* of the given example, which explains why DAGnosis flags this example as inconsistent.

Contrast with Data-SUITE. On the contrary, Data-SUITE doesn't flag this sample since it loses this valuable context because of its compressive representation (e.g. PCA). Furthermore, the walkthrough shown above only makes sense in context of the DAG we learned in Step 2. Hence, Data-SUITE cannot perform Step 4 and Step 5, since it uses a compressive representation, i.e. localization is impossible.

6 DISCUSSION

We have introduced DAGnosis, a data-centric method which leverages structures as representations of data in order to flag inconsistent samples at test-time. We show that structures provide specific and localized information about the consistency of each feature, leading to relevant sample-wise conclusions. We have shown experimentally that this insight provides more accurate detection of inconsistencies and ensures reliable downstream performance. DAGnosis also helps with the understanding of these inconsistencies, by localizing their causes. Future work could build on the insight of the value of structure to advance the data-centric research agenda.

Future Directions. While we have focused in this work on tabular data, other data modalities might benefit from the use of structures to identify and localize inconsistencies, such as time series and natural language. This would require adapting structure discovery and conformal prediction, the two building blocks of DAGnosis.

Acknowledgements

The authors would like to thank Tennison Liu, Hao Sun, Andrew Rashbass and the three anonymous AISTATS reviewers for useful comments on an earlier version of the manuscript. NH is funded by Illumina, JB by the W.D. Armstrong Trust, NS by the Cystic Fibrosis Trust, and JC by Aviva. This work was supported by Azure sponsorship credits granted by Microsoft's AI for Good Research Lab.

References

Asuncion, A. U. (2007). Uci machine learning repository, university of california, irvine, school of information and computer sciences.

Balasubramanian, V. N., Ho, S.-S., and Vovk, V. (2014). Conformal prediction for reliable machine learning: Theory, adaptations and applications.

Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., Garcia, S., Gil-Lopez, S., Molina, D., Benjamins, R., Chatila, R., and Herrera, F. (2020). Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai. *Information Fusion*, 58:82–115.

Bello, K., Aragam, B., and Ravikumar, P. (2022). Dagma: Learning dags via m-matrices and a log-determinant acyclicity characterization. *NeurIPS 2022*.

Berger, C., Paschali, M., Glocker, B., and Kamnitsas, K. (2021). Confidence-based out-of-distribution detection: a comparative study and analysis. In *UNSURE 2021, and 6th International Workshop, PIPPI 2021, Held in Conjunction with MICCAI 2021, Strasbourg, France, October 1, 2021, Proceedings 3*, pages 122–132. Springer.

Berrevoets, J., Kacprzyk, K., Qian, Z., and van der Schaar, M. (2023a). Causal deep learning. *arXiv preprint arXiv:2303.02186*.

Berrevoets, J., Seedat, N., Imrie, F., and Van Der Schaar, M. (2023b). Differentiable and transportable structure learning. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 2206–2233. PMLR.

Borisov, V., Leemann, T., Sessler, K., Haug, J., Pawelczyk, M., and Kasneci, G. (2021). Deep neural networks and tabular data: A survey. *ArXiv*, abs/2110.01889.

Gawlikowski, J., Tassi, C. R. N., Ali, M., Lee, J., Humt, M., Feng, J., Kruspe, A. M., Triebel, R., Jung, P., Roscher, R., Shahzad, M., Yang, W., Bamler, R., and Zhu, X. (2021). A survey of uncertainty in deep neural networks. *ArXiv*, abs/2107.03342.

Geiger, D. and Heckerman, D. (1994). Learning gaussian networks. In *Uncertainty Proceedings 1994*, pages 235–243. Elsevier.

Geiger, D., Verma, T., and Pearl, J. (2013). d-separation: From theorems to algorithms. In *Conference on Uncertainty in Artificial Intelligence*.

Ghosh, S., Yao, J., and Doshi-Velez, F. (2018). Structured variational learning of bayesian neural networks with horseshoe priors. In *International Conference on Machine Learning*, pages 1744–1753. PMLR.

Guyon, I., Aliferis, C., et al. (2007). Causal feature selection. In *Computational methods of feature selection*, pages 79–102. Chapman and Hall/CRC.

Hasan, U. and Gani, M. O. (2022). Kcrl: A prior knowledge based causal discovery framework with reinforcement learning. *Proceedings of Machine Learning Research*, 182(2022):1–24.

Jain, A., Patel, H., Nagalapatti, L., Gupta, N., Mehta, S., Guttula, S., Mujumdar, S., Afzal, S., Sharma Mittal, R., and Munigala, V. (2020). Overview and importance of data quality for machine learning tasks. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3561–3562.

Jordon, J., Yoon, J., and van der Schaar, M. (2018). Knockoffgan: Generating knockoffs for feature selection using generative adversarial networks. In *International conference on learning representations*.

Kalisch, M. and Bühlman, P. (2007). Estimating high-dimensional directed acyclic graphs with the pc-algorithm. *Journal of Machine Learning Research*, 8(3).

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154.

Koller, D. and Friedman, N. (2009). Probabilistic graphical models - principles and techniques.

Krawczyk, B. (2016). Learning from imbalanced data: open challenges and future directions. *Progress in Artificial Intelligence*, 5:221–232.

Lakshminarayanan, B., Pritzel, A., and Blundell, C. (2016). Simple and scalable predictive uncertainty estimation using deep ensembles. In *NIPS*.

Li, Z., Zhao, Y., Botta, N., Ionescu, C., and Hu, X. (2020). Copod: copula-based outlier detection. In *2020 IEEE international conference on data mining (ICDM)*, pages 1118–1123. IEEE.

Liang, W., Tadesse, G. A., Ho, D., Fei-Fei, L., Zaharia, M., Zhang, C., and Zou, J. (2022). Advances, challenges and opportunities in creating data for trustworthy ai. *Nature Machine Intelligence*, 4(8):669–677.

Liu, F. T., Ting, K. M., and Zhou, Z.-H. (2008). Isolation forest. In *2008 eighth ieee international conference on data mining*, pages 413–422. IEEE.

Liu, W., Wang, X., Owens, J., and Li, Y. (2020). Energy-based out-of-distribution detection. *Advances in neural information processing systems*, 33:21464–21475.

Mooij, J. M., Peters, J., Janzing, D., Zscheischler, J., and Schölkopf, B. (2016). Distinguishing cause from effect using observational data: Methods and benchmarks. *Journal of Machine Learning Research*, 17(32):1–102.

Ng, I., Ghassami, A., and Zhang, K. (2020). On the role of sparsity and dag constraints for learning linear dags. *Advances in Neural Information Processing Systems*, 33:17943–17954.

Park, C., Awadalla, A., Kohno, T., and Patel, S. (2021). Reliable and trustworthy machine learning for health using dataset shift detection. *Advances in Neural Information Processing Systems*, 34:3043–3056.

Peters, J., Janzing, D., and Schölkopf, B. (2017). Elements of causal inference: Foundations and learning algorithms.

Pinna, A., Heise, S., Flassig, R. J., Fuente, A. d. l., and Klamt, S. (2013). Reconstruction of large-scale regulatory networks based on perturbation graphs and transitive reduction: improved methods and their evaluation. *BMC systems biology*, 7:1–19.

Polyzotis, N., Roy, S., Whang, S. E., and Zinkevich, M. (2017). Data management challenges in production machine learning. In *Proceedings of the 2017 ACM International Conference on Management of Data*, SIGMOD '17, page 1723–1726, New York, NY, USA. Association for Computing Machinery.

Rasmussen, C. E. and Williams, C. K. I. (2003). Gaussian processes for machine learning. In *Adaptive computation and machine learning*.

Renggli, C., Rimanic, L., Gürel, N. M., Karlas, B., Wu, W., and Zhang, C. (2021). A data quality-driven view of mlops. *IEEE Data Engineering Bulletin*.

Romano, Y., Patterson, E., and Candès, E. J. (2019). Conformalized quantile regression. In *Neural Information Processing Systems*.

Sachs, K., Perez, O. D., Pe'er, D., Lauffenburger, D. A., and Nolan, G. P. (2005). Causal protein-signaling networks derived from multiparameter single-cell data. *Science*, 308:523 – 529.

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., and Aroyo, L. M. (2021). "everyone wants to do the model work, not the data work": Data cascades in high-stakes ai. In *proceedings of the*

2021 CHI Conference on Human Factors in Computing Systems, pages 1–15.

Saria, S. and Subbaswamy, A. (2019). Tutorial: Safe and reliable machine learning. *ACM Conference on Fairness, Accountability, and Transparency*.

Seedat, N., Crabbe, J., and Schaar, M. (2022a). Datasuite: Data-centric identification of in-distribution incongruous examples. In *International Conference on Machine Learning*. PMLR.

Seedat, N., Imrie, F., and van der Schaar, M. (2022b). Dc-check: A data-centric ai checklist to guide the development of reliable machine learning systems. *arXiv preprint arXiv:2211.05764*.

Sinha, M. and Ramsey, S. A. (2021). Using a general prior knowledge graph to improve data-driven causal network learning. In *AAAI Spring Symposium: Combining Machine Learning with Knowledge Engineering*.

Spirtes, P., Glymour, C. N., Scheines, R., and Heckerman, D. (2000). *Causation, prediction, and search*. MIT press.

Torralba, A. and Efros, A. A. (2011). Unbiased look at dataset bias. *CVPR 2011*, pages 1521–1528.

Verma, T. and Pearl, J. (1990a). Causal networks: Semantics and expressiveness. In *Machine intelligence and pattern recognition*, volume 9, pages 69–76. Elsevier.

Verma, T. S. and Pearl, J. (1990b). Equivalence and synthesis of causal models. In *Proceedings of the Sixth Conference on Uncertainty in Artificial Intelligence*.

Vovk, V., Gammerman, A., and Shafer, G. (2005). Conformal prediction. *Algorithmic learning in a random world*, pages 17–51.

Wu, Y. and Verdú, S. (2011). Functional properties of minimum mean-square error and mutual information. *IEEE Transactions on Information Theory*, 58(3):1289–1301.

Yang, J., Lindenbaum, O., and Kluger, Y. (2022). Locally sparse neural networks for tabular biomedical data. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 25123–25153. PMLR.

Yeh, I.-C. and Lien, C.-h. (2009). The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. *Expert systems with applications*, 36(2):2473–2480.

Yuksekgonul, M., Zhang, L., Zou, J., and Guestrin, C. (2023). Beyond confidence: Reliable models should also consider atypicality. *arXiv preprint arXiv:2305.18262*.

Zhao, Y., Hu, X., Cheng, C., Wang, C., Wan, C., Wang, W., Yang, J., Bai, H., Li, Z., Xiao, C., et al. (2021). Suod: Accelerating large-scale unsupervised heterogeneous outlier detection. *Proceedings of Machine Learning and Systems*, 3:463–478.

Zheng, X., Aragam, B., Ravikumar, P., and Xing, E. P. (2018). Dags with no tears: Continuous optimization for structure learning. In *NeurIPS*.

Zheng, X., Dan, C., Aragam, B., Ravikumar, P., and Xing, E. P. (2020). Learning sparse nonparametric dags. *ArXiv*, abs/1909.13189.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes] See Section 3
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Yes] Included in the supplementary material.
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [No] Code will be released upon publication.

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Not Applicable]
   (b) Complete proofs of all theoretical results. [Not Applicable]
   (c) Clear explanations of any assumptions. [Not Applicable]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes] Link to code is provided.
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Yes] Included in the supplementary material.

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]

Appendix: DAGnosis: Localized Identification of Data Inconsistencies using Structures

A ADDITIONAL DETAILS ON DAGNOSIS

A.1 Conformal Prediction

In this section, we give additional details regarding the conformal prediction part underpinning DAGnosis.

A.1.1 Bonferroni Correction

In our experiments, we flag a sample x if $\nu(x) > 0$, i.e. at least one of the features of x is inconsistent. This is similar to a multiple hypothesis testing setup, since we are computing $\nu_i(x)$ for $i \in [d]$ in order to define $\nu(x)$. In order to control the FWER (Family Wise Error Rate, or the probability to make a false discovery when multiple tests are conducted), we adopt the Bonferroni correction, which, for a desired significance level α_0 consists of setting the following significance level for the individual tests: $\alpha = \frac{\alpha_0}{d}$. Given this, the union bound leads to $FWER \leq \sum_{i=1}^{d} \frac{\alpha_0}{d} = \alpha_0$.

A.1.2 Calibration Step in CQR

Suppose that lower and upper quantile estimators $\hat{q}_{i,\alpha_{lo}}$ and $\hat{q}_{i,\alpha_{hi}}$ have been trained on a proper training set $\mathcal{D}^{+}_{\text{train}}$. The calibration step is critical in order to ensure that the confidence intervals ensure coverage guarantees, and it relies on a calibration set.

Let us fix a calibration set $\mathcal{D}_{\text{cal}} = \{x^{(1)}, ..., x^{(n_{cal})}\}$. Following Romano et al. (2019), and for $i \in [d]$, we compute the set of non-conformity scores $C_i = \{E_{i,j} \mid j \in [n_{cal}]\}$ where $E_{i,j} := \max\{\hat{q}_{i,\alpha_{lo}}(x^{(j)}) - x_i^{(j)}, x_i^{(j)} - \hat{q}_{i,\alpha_{hi}}(x^{(j)})\}$

Next, we compute $\epsilon_{\alpha,i}$, the $(1-\alpha)(1+\frac{1}{n_{cal}})$-th empirical quantile of C_i, which is used to construct the prediction intervals, with $l_{i,\alpha} := \hat{q}_{i,\alpha_{lo}} - \epsilon_{\alpha,i}$ and $r_{i,\alpha} := \hat{q}_{i,\alpha_{hi}} + \epsilon_{\alpha,i}$.

A.1.3 Coverage Guarantee

A compelling property of conformal prediction is the marginal coverage guarantee, which holds under the exchangeability assumption (a sequence of random variables is said to be exchangeable if any permutation of the sequence has the same joint probability distribution as the original sequence). We state the marginal coverage property stemming from this assumption:

Proposition A.1 (Marginal coverage)**.** *If the data points $X^{(1)}, X^{(2)}, ..., X^{(n+1)}$ are exchangeable (with $\mathcal{D}_{cal}$ defined as the set comprising the first n samples) , we have for all $i \in [d]$ that $\mathbb{P}(X_i^{(n+1)} \in [l_{i,\alpha}(X^{(n+1)}), r_{i,\alpha}(X^{(n+1)})]) \geq 1 - \alpha$ for $0 < \alpha < 1$.*

Note that the calibration set is implicitly used to define $l_{i,\alpha}$ and $r_{i,\alpha}$. The marginal coverage guarantee is appealing, because it permits to control a desired False Positive Rate for inconsistency detection.

A.1.4 Quantile Regression

In our experiments, we use a LightGBM model (Ke et al., 2017) as the quantile regression backbones in CQR. In order to tune the hyperparameters of all the methods, we perform a random search with $n_{\text{iter}} = 100$, to tune the number of leaves (range (10,50)), the maximum depth (range (3, 20)), the number of estimators (range (50, 300)), and the learning rate (range (0,1)). Moreover, we perform K-fold cross-validation with 5 folds. We fit the lower and upper quantile regressors using the α-pinball loss, defined by:

$$\rho_\alpha(x, x') := \begin{cases} \alpha(x - x') & \text{if } x - x' > 0, \\ (1-\alpha)(x' - x) & \text{otherwise} \end{cases} \tag{2}$$

A.2 Inference with DAGnosis

We summarize in Algorithm 2 how DAGnosis is used at test time to flag inconsistencies. DAGnosis outputs a confidence interval for any feature i of a sample x. When x_i does not fall inside this confidence interval, the feature i of x is deemed inconsistent.

Algorithm 2 – Inference. Using a trained DAGnosis model (cfr. Algorithm 1), we describe how one can test the samples in $\mathcal{D}_{\text{test}}$ for inconsistencies.

Input: A list of conformal predictors $\{[l_{i,\alpha}, r_{i,\alpha}] \mid i \in [d]\}$, a testing set $\mathcal{D}_{\text{test}}$, an empty set of inconsistent samples $\mathcal{D}_{\text{test,incons}} = \varnothing$
Output: Updated set of inconsistent samples $\mathcal{D}_{\text{test,incons}}$
for $x \in \mathcal{D}_{\text{test}}$ **do**
  for $i \in [d]$ **do**
    if $x_i \notin [l_{i,\alpha}(x), r_{i,\alpha}(x)]$ **then**
      Add x to inconsistent samples, $\mathcal{D}_{\text{test,incons}}$
    end if
  end for
end for

A.3 Markov Boundaries (MB) Should be Preferred Over Parents

We now explain why DAGnosis uses $MB(X)$ rather than $Pa(X)$ to flag inconsistencies. We do so both empirically and theoretically.

A.3.1 Empirical Demonstration

We provide additional results comparing between using MB and Parents in a synthetic setup with MLP SEMs, for the task of detecting inconsistencies, with $d = 20$, $s = 20$. We report in Table 3 the mean and standard errors of the F1 score, precision and recall for 20 runs. As we can see, using the MB leads to a more accurate detection of inconsistencies than solely using the parents of each feature.

Table 3: **Markov Boundaries Capture the Relevant Information.** Comparison between conditioning on the markov boundaries (MB) or the parents nodes only, for $d = 20$, $s = 20$. Mean and standard error for the F1 score, precision and recall for the inconsistency detection task are reported (↑ is better)

	F1	Precision	Recall
Parents	0.92 ± 0.02	0.91 ± 0.01	0.94 ± 0.03
MB	$\mathbf{0.94} \pm 0.01$	$\mathbf{0.92} \pm 0.01$	$\mathbf{0.96} \pm 0.02$

A.3.2 Theoretical Justification

A markov boundary of X_i defines the minimal set of features which contains all the information relevant to predict X_i. This can be stated in terms of conditional independence (Def. 4.11 in Koller and Friedman (2009) and Def. 3.1 in the main paper) or equivalently mutual information, i.e. $I(X_i, S \setminus S_i | S_i) = 0$, where S_i is a MB of X_i. Furthermore, MB are minimal for this property. We emphasize that $Pa(X_i)$ does not necessarily satisfy the CI property, because $Pa(X_i)$ ignores informative nodes. When regressing X_i with the mean squared error, $I(X_i, S \setminus S_i | S_i) = 0$ implies that the optimal regressor can be expressed as a function of the MB S_i (see Wu and Verdú (2011)).

A.4 Differences Between DAGnosis and OOD Detection

We now emphasize the differences between DAGnosis and OOD detectors.

A.4.1 Most OOD Detectors are not Widely Applicable and not Tailored to Tabular Data

Energy-based OOD detection (Liu et al., 2020) and confidence-based detection (Berger et al., 2021) encompass a big proportion of the OOD detection litterature. However, they most often rely on neural networks, yet we

are interested in tabular data, where tree-based methods are prevalent. Furthermore, they also assume access to labels, which is not required by DAGnosis. Indeed, recall that DAGnosis operates at a feature level, by constructing feature-wise confidence intervals.

A.4.2 DAGnosis Brings Localization, OOD Detectors do not

DAGnosis permits a fine-grained analysis of samples with feature-wise confidence intervals. This fundamental novelty separates DAGnosis from OOD detectors, which traditionally adopt a generative approach based on estimating a (joint) likelihood $P(X)$. They impose a threshold on the likelihood to separate inliers from outliers. However, this does not give an explanation as to where inconsistencies happen, i.e. there is no localization.

A.4.3 Additional Experiment

We compare experimentally DAGnosis to several OOD detectors tailored for the tabular domain: SUOD (Zhao et al., 2021), Iforest (Liu et al., 2008), and COPOD (Li et al., 2020). We compute in a synthetic setup the proportion of samples flagged by each of these detectors which are also flagged by DAGnosis, for the real-world dataset **UCI Adult Income**. We also report the downstream accuracy when we defer prediction on the samples detected as inconsistent. We report the results in Table 4 and Table 5. We conclude that DAGnosis detects a more fine-grained class of inconsistent samples (i.e. in-distribution inconsistencies) thanks to localization, which are harmful for downstream tasks. As such, the work most related to ours is the SOTA Data-SUITE (Seedat et al., 2022a), which also deals with ID inconsistencies.

Table 4: **DAGnosis Differs From OOD Detectors.** Proportion of the samples returned by the OOD detectors which are also flagged by DAGnosis (we set the thresholds such that each method flags the same number of samples)

	COPOD	**Iforest**	**SUOD**
Overlap proportion	0.35	0.38	0.45

Table 5: **DAGnosis Flags Harmful Inconsistent Examples.** Downstream accuracy (↑ is better) evaluated for the samples deemed consistent by each method (complement of the inconsistent samples in the test set). Note that the thresholds of COPOD, Iforest and SUOD are set such that every method (including DAGnosis) flags the same number of inconsistencies for a fair comparison.

	DAGnosis	**COPOD**	**Iforest**	**SUOD**
Downstream accuracy	0.701	0.654	0.689	0.679

B DETAILS ON THE EXPERIMENTAL SETUP

All the experiments were run on a machine equipped with a 64-Core AMD Ryzen Threadripper and a NVIDIA RTX A4000.

B.1 Synthetic Setup

B.1.1 Generation of the Synthetic Data

We give more details here on the setup used to generate the synthetic data in Section 4.1.

▪ **DAG et SEM sampling** The DAGs are sampled following the Erdős–Rényi model (d, s), which means that they are sampled uniformly in the set of DAGs containing d nodes and s edges. These DAGs define sets of parents, where $Par(i)$ is the set of parents of feature X_i.

Linear setting For $i \in [d]$, we consider the SEMs defined by $X_i = W_i^T X_{Par(i)} + Z_i$, with $Z_i \sim \mathcal{N}(0, 1)$ a gaussian noise independent of $X_{Par(i)}$ and $W_i \in \mathbb{R}^{|Par(i)|}$. Each dimension in the parameter W_i is sampled following a mixture of the uniform distributions $\mathcal{U}(-2.5, -0.5)$ and $\mathcal{U}(0.5, 2.5)$, with weights 0.5 and 0.5, following Zheng et al. (2018).

MLP setting For $i \in [d]$, we consider the SEMs defined by $X_i = {W_i^1}^T \sigma(W_i^2 X_{Par(i)}) + Z_i'$, with $Z_i' \sim \mathcal{N}(0, 1)$ a gaussian noise independent of $X_{Par(i)}$ and σ is the sigmoid function. Moreover, $W_i^2 \in \mathbb{R}^{h \times |Par(i)|}$ and $W_i^1 \in \mathbb{R}^h$, with $h = 100$. Each parameter in W_i^1 and W_i^2 is sampled following a mixture of $\mathcal{U}(-2.5, -0.5)$ and $\mathcal{U}(0.5, 2.5)$, with weights 0.5 and 0.5.

▪ **Data splitting** The training set $\mathcal{D}_{\text{train}}$ is split into a proper training set $\mathcal{D}_{\text{train}}^+$ and a calibration set $\mathcal{D}_{\text{cal}}$, with $\frac{|\mathcal{D}_{\text{cal}}|}{|\mathcal{D}_{\text{train}}|} = 0.2$.

▪ **Generation of corruptions** In Section 4.1, we corrupt the SEMs both in the linear and the MLP settings. We define these corruptions as follows.

Linear setting We consider the perturbed parameters $W_i' = W_i + U_i$, with $U_i \sim \mathcal{N}(m\mathbf{1}, I)$. We take $m = 5$. W_i' then replaces the W_i in the definition of the SEMs.

MLP setting We consider the perturbed parameters ${W'}_i^1 = W_i^1 + V_i \odot M$, with $V_i \sim \mathcal{N}(m\mathbf{1}, I)$, and M a random binary mask (in Section 4.1, we have $\sum_{i=1}^h M_i = 5$). Hence we perturb the last layer of the MLP. We take $m = 2$. ${W'}_i^1$ then replaces W_i^1 in the definition of the SEMs.

▪ **DAG discovery with NOTEARS** We describe the protocol we follow to discover the DAGs. We use the differentiable structure learner NOTEARS, which comes with two variants: NOTEARS Linear (Zheng et al., 2018) and NOTEARS MLP (Zheng et al., 2020).

NOTEARS Linear We use the following parameters: maxiter $= 100$, $h_{\text{tol}} = 10^{-8}$, $\rho_{\max} = 10^{16}$, $w_{\text{threshold}} = 0.3$ (adjusted when the resulting graph is not a DAG).

NOTEARS MLP We use the following parameters: $h_{\text{tol}} = 10^{-10}$, $\rho_{\max} = 10^{18}$, hidden dimension of the MLP $= 10$, $\lambda_1 = 0.01$, $w_{\text{threshold}} = 0.3$ (adjusted when the resulting graph is not a DAG).

B.2 UCI Adult Income Dataset (Asuncion, 2007)

The UCI Adult income dataset was extracted from the 1994 Census bureau database. It is licensed under a Creative Commons Attribution 4.0 International.

B.2.1 Controlling the Inconsistencies in the Train/Test Split

As described in Section 4.4, we construct specific train/test splits to control the amount of inconsistencies. Let us denote $\mathcal{D}_{\text{S=0,I=0}}$ the set of samples having *Sex = 0 (Women), Income* $\leq$ *50k* (in what follows, *Sex* is abbreviated to S, *Income* to I, and the subscripts denote the values taken by these features). Furthermore, for $k \in [9]$ we consider a list of k datasets, $\mathcal{D}_{\text{S=0,I=0}}^{(k)} \subset \mathcal{D}_{\text{S=0,I=0}}$, with $|\mathcal{D}_{\text{S=0,I=0}}^{(k)}| = 1000k$. Given these notations, let us now define $\mathcal{D}_{\text{train}} = \mathcal{D}_{\text{train,S=1}} \bigsqcup \mathcal{D}_{\text{S=0,I=1}}$ and $\mathcal{D}_{\text{test}}^{(k)} = \mathcal{D}_{\text{test,S=1}} \bigsqcup \mathcal{D}_{\text{S=0,I=0}}^{(k)}$, with $|\mathcal{D}_{\text{train,S=1}}| = |\mathcal{D}_{\text{test,S=1}}|$. In a nutshell, we

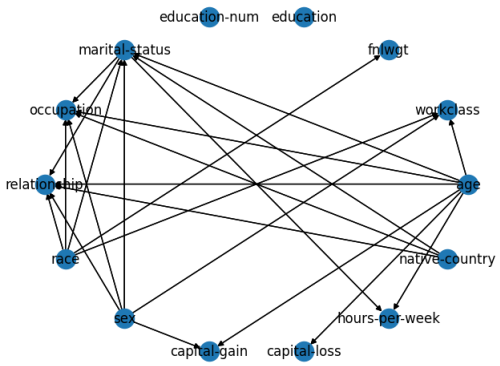


Figure 5: **Adult DAG.** DAG discovered with the PC algorithm, using $\mathcal{D}_{\text{train}}$.

put women with high income in the training set, and control the number of women with low income in the test set.

B.2.2 DAG Discovery

There is no ground-truth structure provided with the UCI Adult income dataset. Hence we need to discover the DAG. In order to discover the DAG, we use the PC algorithm (Spirtes et al., 2000). We also split the features into three different tiers, which define a set of forbidden edges.

- Tier 1: Age, Sex, Race, Native-country
- Tier 2: Education, Education-num, Marital-status
- Tier 3: all the other features

We use the Chi-squared conditional independence test, with a significance level of 0.01, after having binned the continuous variables. The discovered DAG is depicted in Figure 5. Note that we discover the DAG using only $\mathcal{D}_{\text{train}}$, which is not the full dataset, to avoid any data leakage.

B.2.3 Downstream Classifier

We consider a random forest classifier as our downstream model, with $n_{\text{estimators}} = 100$.

C ADDITIONAL RESULTS

C.1 Sensitivity with Respect to the Discovered DAG in 4.1

Methodology. In Section 4.1, we discover the DAG underlying the synthetic datasets by leveraging NOTEARS, and the DAG is then used by DAGnosis. A natural question one could ask is how sensitive the approach is to the discovered DAG. In order to answer this question, in the MLP SEMs setting, we plot the average Structural Hamming Distance of the discovered DAGs to the ground-truth DAG (used to generate the data), which is the minimal number of edits to go from the ground-truth DAG to the discovered DAG. This takes into account edge removals, edge additions, and edge reversals.

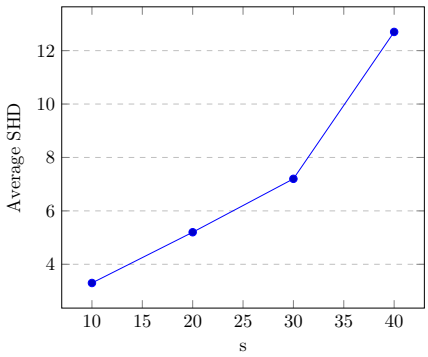


Figure 6: **Average SHD.** We report the average Structural Hamming Distance of the discovered DAGs with respect to the ground-truth DAG, as a function of the real number of edges in this ground-truth DAG, in the MLP SEMs setting.

Results. As we can see in Figure 6, the average SHD is an increasing function of the number of edges in the ground-truth DAG, which is intuitive, since a higher density implies a harder setting for DAG discovery. However, what is striking is that the results obtained by DAGnosis NOTEARS in Section 4.1 almost match those of DAGnosis GT. This robustness result is a desideratum in real-world settings, where the underlying DAG is often unknown and needs to be discovered. These results corroborate the robustness highlighted in Section 4.2, where we investigate the impact of corrupting the DAGs provided to DAGnosis.

C.2 AUROC Results

Methodology. In addition to the F1-scores, precision and recall metrics reported in Section 4.1, we also compute an AUROC (Area Under the Receiver Operating Characteristic) to compare DAGnosis with Data-SUITE across different significance levels. In order to construct the ROC, we sweep the significance level α across $\{\frac{0.1k}{d} \mid k \in [10]\}$. We then train the conformal estimators with each value of α, which permits to compute at test-time a False Positive rate and a True Positive rate. The AUROC is then approximated using the trapezoidal rule. We report in Table 2 the mean and $1.96 \times SE$, where SE denotes the standard error, for 5 DAGs per $s \in \{10, 20, 30, 40\}$, in the setting with MLP SEMs.

Table 6: **AUROC** We report the AUROC of the different methods, in the synthetic setting of MLP SEMs, where 5 DAGs are sampled for each s. DAGnosis NOTEARS consistently outperforms the two baselines, highlighting the importance of structures

	$s = 10$	$s = 20$	$s = 30$	$s = 40$
DN Auto	0.93 (.01)	0.94 (.01)	0.94 (.00)	0.91 (.01)
Data-SUITE CQR	0.93 (.01)	0.95 (.00)	0.89 (.03)	0.91 (.02)
DN (NT)	0.95 (.00)	0.96 (.00)	0.95 (.00)	0.94 (.01)

Results. As we can see in Table 2, DAGnosis NOTEARS consistently outperforms the baselines, which corroborates the findings of Section 4.1, and illustrates the importance of incorporating the rich information given by structures.

C.3 High-dimensional Results

Methodology. We complement Section 4.2 in the main paper with additional results in high-dimension. For this, we set $d = 200$, which is a 10 time increase to the dimension of the data used in Section 4.1. We also set $s = 200$ and $n = 20000$, and consider DAGnosis PC versus Data-SUITE. We report the mean and standard errors of the F1 score, precision and recall for 10 runs in Table 7, for the task of detecting inconsistencies, similarly to Section 4.1.

Results. As we can see in Table 7, DAGnosis outperforms the SOTA Data-SUITE by a large margin in high-dimensional settings, on all the detection metrics.

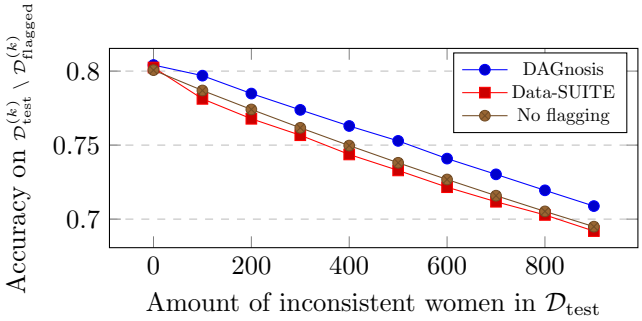
Table 7: **DAGnosis also Outperforms the SOTA in High Dimension.** High-dimensional results for the task of detecting inconsistencies, $d = 200$, $s = 200$, $n = 20000$. Mean and standard error for the F1 score, precision and recall for the inconsistency detection task are reported (↑ is better)

	F1 score	Precision	Recall
Data-SUITE	0.48 ± 0.11	0.70 ± 0.05	0.36 ± 0.10
DAGnosis	**0.71** ± 0.09	**0.86** ± 0.03	**0.61** ± 0.11

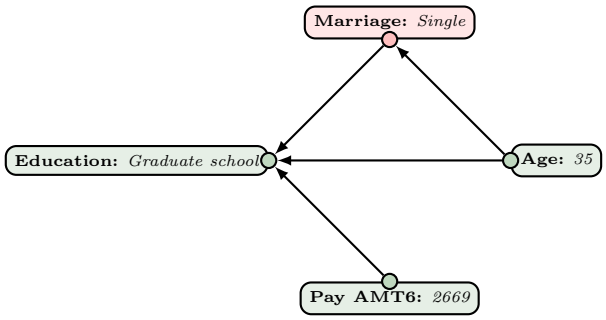
C.4 Additional Experiment for Reliable Downstream Performance

Methodology. In this experiment, we aim to show that DAGnosis enables reliable downstream performance when deferring prediction on the set of inconsistent samples, similarly to Section 4.4. We use the dataset **Credit**, which is a financial default dataset from a Taiwan bank (Yeh and Lien, 2009), with $d = 23$. We define specific train/test splits to control the presence of inconsistencies in the test dataset. More precisely, we split men equally in $\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{test}}$. We put women with default payment in $\mathcal{D}_{\text{train}}$. We gradually add women without default payment in $\mathcal{D}_{\text{test}}$, with a parameter k controlling the number of such samples, giving a list of k test sets $\mathcal{D}^{(k)}_{\text{test}}$. This motivates identifying inconsistencies in $\mathcal{D}^{(k)}_{\text{test}}$ with respect to $\mathcal{D}_{\text{train}}$, as we expect to flag most of the women in $\mathcal{D}^{(k)}_{\text{test}}$ as inconsistent. We learn the DAG using the PC algorithm.

Results. We then compute the downstream task accuracy on $\mathcal{D}^{(k)}_{\text{test}} \setminus \mathcal{D}^{(k)}_{\text{flagged}}$, i.e. we defer prediction on the inconsistent samples. As shown in Figure 7a, DAGnosis flags samples which are more harmful for the downstream task than samples flagged by Data-SUITE, evidenced by the higher downstream accuracy on $\mathcal{D}^{(k)}_{\text{test}} \setminus \mathcal{D}^{(k)}_{\text{flagged}}$. We also show in Figure 7b an example of a sample flagged by DAGnosis. It illustrates the localization property inherent to our method, which contrasts Data-SUITE – a method incapable of providing such localization because it uses compressive representations.



(a) Downstream accuracy for non-flagged samples, for the Credit dataset



(b) We depict the Markov boundary for the feature *Marriage*, which is flagged by DAGnosis for the given example.

Figure 7: **(a):** Deferring prediction on $\mathcal{D}^{(k)}_{\text{flagged}}$, the set of samples flagged by DAGnosis, leads to a better downstream accuracy. **(b):** DAGnosis provides localization. This localization contrasts Data-SUITE which uses compressive representations.

D STRUCTURE LEARNING OVERHEAD

In this section, we address the questions revolving around structure learning and its time/computation overhead, which is inherently defined by the structure learning method used. While structure learning is difficult in high-dimensional setups and is not an easy task in the real world, there are several reasons which make it doable as part of DAGnosis.

Structure Learning is Conducted Once. We emphasize that structure learning is a step which is taken independently from feature-wise conformal prediction. This step is conducted only once for a given training dataset, and can be done in parallel to training the model in the first place. Moreover, DAGnosis is very flexible in the way the structure is provided. Indeed, when the structure needs to be discovered, DAGnosis is completely agnostic to the structure learner. We illustrate this with three structure learners which are the PC algorithm, NOTEARS, and DAGMA.

Recent Advances in Structure Learning. The flexibility in the way the structure is learnt is a key advantage of DAGnosis, because it permits to take full advantage of the recent advances in structure learning, which have made the cost of learning structures completely bearable. As an example, we refer to DAGMA (Bello et al., 2022), and especially Figures 4 and 5 in the corresponding paper, which illustrate the number of dimensions for which the method can be scaled up to, and the runtime. As we can see, in its linear version, this structure learner can tackle dimensions up to 1000, in a reasonable time, while maintaining good discovery accuracy.

E BROADER IMPACT

The research presented in this work on the identification and handling of inconsistencies in data at deployment time holds significant broader impacts for the machine learning community. By addressing the limitations of existing data-centric methods DAGnosis brings valuable insights and advancements to the field.

One of the key broader impacts of DAGnosis is its potential to enhance the reliability and trustworthiness of machine learning models. Inconsistencies in data can significantly impact the performance of models, leading to potential errors and misinterpretations in real-world applications. By leveraging structural interactions, DAGnosis enables more accurate conclusions in detecting inconsistencies, thereby improving the overall reliability of machine learning models in practical settings. It is vital that inconsistency detection does not create additional bias in the real world. First, we stress that DAGnosis assesses $\mathcal{D}_{\text{test}}$ with respect to the reference dataset $\mathcal{D}_{\text{train}}$, which is typically assumed to be a representative dataset of the distribution of interest. Second, DAGnosis provides a safeguard against erroneous inconsistency detection with the marginal coverage guarantee under exchangeability.

Another significant broader impact of this research is the localization of causes of inconsistencies on a DAG. Previous approaches have often lacked the ability to pinpoint the specific reasons why a sample might be flagged as inconsistent. However, DAGnosis addresses this limitation by providing detailed insights into the factors contributing to inconsistencies. This localization capability not only helps in understanding and interpreting the flagged samples but can also guide future data collection. Researchers and practitioners can use this information to refine data collection strategies, thereby improving the quality of training data.

By addressing the limitations of existing approaches, DAGnosis paves the way for more accurate and insightful data-centric conclusions.