Causal Inference on Multivariate and Mixed-Type Data

Alexander Marx and Jilles Vreeken

Max Planck Institute for Informatics and Saarland University, Saarbrücken, Germany
{amarx,jilles}@mpi-inf.mpg.de

Abstract. How can we discover whether X causes Y, or vice versa, that Y causes X, when we are only given a sample over their joint distribution? How can we do this such that X and Y can be univariate, multivariate, or of different cardinalities? And, how can we do so regardless of whether X and Y are of the same, or of different data type, be it discrete, numeric, or mixed? These are exactly the questions we answer. We take an information theoretic approach, based on the Minimum Description Length principle, from which it follows that first describing the data over *cause* and then that of *effect* given *cause* is shorter than the reverse direction. Simply put, if Y can be explained more succinctly by a set of classification or regression trees conditioned on X, than in the opposite direction, we conclude that X causes Y. Empirical evaluation on a wide range of data shows that our method, CRACK, infers the correct causal direction reliably and with high accuracy on a wide range of settings, outperforming the state of the art by a wide margin.

1 Introduction

Telling cause from effect is one of the core problems in science. It is often difficult, expensive, or impossible to obtain data through randomized trials, and hence we often have to infer causality from, what is called, observational data [19]. We consider the setting where, given data over the joint distribution of two random variables X and Y, we have to infer the causal direction between X and Y. In other words, our task is to identify whether it is more likely that X causes Y, or vice versa, that Y causes X, or that the two are merely correlated.

In practice, X and Y do not have to be of the same type. The altitude of a location (real-valued), for example, determines whether it is a good habitat (binary) for a mountain hare. In fact, neither X nor Y have to be univariate. Whether or not a location is a good habitat for an animal, is not just caused by a single aspect, but by a *combination* of conditions, which not necessarily are of the same type. We are therefore interested in the general case where X and Y may be of any cardinality, and may be single or mixed-type.

To the best of our knowledge there exists no method for this general setting. Causal inference based on conditional independence tests, for example, requires three variables, and cannot decide between $X \to Y$ and $Y \to X$ [19]. All existing

methods that consider two variables are only defined for single-type pairs. Additive Noise Models (ANMs), for example, have only been proposed for univariate pairs of real-valued [20] or discrete variables [21], and similarly so for methods based on the independence of $P(X)$ and $P(Y \mid X)$ [25]. Trace-based methods require both X and Y to be strictly multivariate real-valued [10,4], and whereas ERGO [31] also works for univariate pairs, these again have to be real-valued. We refer to Sec. 3 for a more detailed overview of related work.

Our approach is based on algorithmic information theory. That is, we follow the postulate that if $X \to Y$, it will be easier—in terms of Kolmogorov complexity—to first describe X, and then describe Y given X, than the inverse direction [11,31,3]. Kolmogorov complexity is not computable, but can be approximated through the Minimum Description Length (MDL) principle [23,7], which we use to instantiate this framework. In addition, we develop a new causal indicator that is able to handle multivariate and mixed-type data.

In particular, we we define an MDL score for coding forests, a model class where a model consists of classification and regression trees as this allows us to consider both discrete and continuous-valued data with one unified model. By allowing dependencies from X to Y, or vice versa, we can measure the difference in complexity between $X \to Y$ and $Y \to X$. Discovering a single optimal decision tree is already NP-hard, and hence we cannot efficiently discover the coding forest that describes the data most succinctly. We therefore propose CRACK, an efficient greedy algorithm for discovering good models directly from data. The inferences we make hence are all with respect to the class of coding trees, and the specific encoding we define. We discuss the implications with regard to identifiability, and through extensive empirical evaluation on synthetic, benchmark, and real-world data, we show that CRACK performs very well in practice—even under adversarial settings.

Our main contributions are as follows. We introduce the first framework for inferring the causal direction from univariate and multivariate single and mixed-type data—as opposed to existing methods that are only able to deal with either nominal or numeric data. We propose a new causal indicator based on the algorithmic Markov condition, instantiate it through MDL, and propose a fast algorithm to compute it. We provide extensive empirical evaluation of our method, in which we additionally introduce new multivariate cause-effect pairs with known ground truth. The paper is organized as usual.

2 Preliminaries

In this section, we introduce the notation and give brief primers to Kolmogorov complexity and the Minimum Description Length principle.

2.1 Notation

In this work we consider two sets of random variables X and Y. Further, we are given a data set D containing n i.i.d. samples drawn from the joint distribution of

X and Y. For convenience, we call A the set of all random variables, where $A = X \cup Y$, with $|A| = m$ being the number of random variables in A. In the following, we will often refer to a random variable $A_i \in A$ as an attribute, regardless of whether it belongs to X or Y. An attribute A_i has a type, where $\text{type}(A_i) \in \{\textit{binary}, \textit{categorical}, \textit{numeric}\}$. We will refer to binary and categorical attributes as *nominal* attributes. We write $\mathcal{X}_i$ to denote the domain of an attribute A_i. Respectively, the size of the domain of an attribute $|\mathcal{X}_i|$ is for discrete data simply the number of distinct values and for numeric data equal to $\frac{\max(A_i) - \min(A_i)}{\text{res}(A_i)} + 1$, where $\text{res}(A_i)$ is the resolution at which the data over attribute A_i was recorded. For example, a resolution of 1 means that we consider integers, of 0.01 means that A was recorded with a precision of up to a hundredth.

We will consider decision and regression trees. A tree T consist of $|T|$ nodes. We identify internal nodes as $v \in \text{int}(T)$, and leaf nodes as $l \in \text{lvs}(T)$. A leaf node l contains $|l|$ data points. All logarithms are to base 2, and we use $0 \log 0 = 0$.

2.2 Kolmogorov Complexity, a brief primer

The Kolmogorov complexity of a finite binary string x is the length of the shortest binary program p^* for a universal Turing machine $\mathcal{U}$ that generates x, and then halts [15]. Formally, we have $K(x) = \min\{|p| \mid p \in \{0,1\}^*, \mathcal{U}(p) = x\}$. Simply put, p^* is the most succinct *algorithmic* description of x, and the Kolmogorov complexity of x is the length of its ultimate lossless compression. Conditional Kolmogorov complexity, $K(x \mid y) \leq K(x)$, is then the length of the shortest binary program p^* that generates x, and halts, given y as input.

By definition, Kolmogorov complexity will make maximal use of any structure in x that can be expressed more succinctly algorithmically than by printing it verbatim. As such it is the theoretical optimal measure for complexity. However, due to the halting problem it is not computable [15]. Instead, we can approximate it from above through MDL [15].

2.3 MDL, a brief primer

The Minimum Description Length (MDL) principle [23,7] is a practical variant of Kolmogorov Complexity. Intuitively, instead of all programs, it considers only those programs that we know that output x and halt. Formally, given a model class $\mathcal{M}$, MDL identifies the best model $M \in \mathcal{M}$ for data D as the one minimizing $L(D, M) = L(M) + L(D \mid M)$, where $L(M)$ is the length in bits of the description of M, and $L(D \mid M)$ is the length in bits of the description of data D given M. This is known as two-part MDL. There also exists one-part, or *refined* MDL, where we encode data and model together. Refined MDL is superior in that it avoids arbitrary choices in the description language L, but in practice only computable for certain model classes. Given infinite data the model costs degenerate to an additive constant term, which is independent of the data. Hence given infinite data, two-part MDL converges to refined MDL. Note that in either case we are only concerned with code *lengths*—our goal is to measure the *complexity* of a dataset under a model class, not to actually compress it [7].

3 Related Work

Causal inference on observational data is a challenging problem, and has recently attracted a lot of attention [19,11,26,3]. Most existing proposals, are highly specific in the type of causal dependencies and type of variables they can consider.

Classical constrained-based approaches, such as conditional independence tests, require three observed random variables [27,19], cannot distinguish Markov equivalent causal DAGs [30] and hence cannot decide between $X \to Y$ and $Y \to X$. Recently, there has been increased attention for methods that can infer the causal direction from only two random variables. Generally, they exploit certain properties of the joint distribution.

Additive Noise Models (ANMs) [26], for example, assume that the effect is a function of the cause and cause-independent additive noise. ANMs exist for univariate real-valued [26,9,32,20] and discrete data [21]. It is unclear how to extend this model for multivariate or mixed type data. A related approach considers the asymmetry in the joint distribution of *cause* and *effect* for causal inference. The linear trace method (LTR) [10] and the kernelized trace method (KTR) [4] aim to find a structure matrix A and the covariance matrix Σ_X to express Y as AX. Both methods are restricted to multivariate continuous data. In addition, KTR assumes a deterministic, functional and invertible causal relation. Sgouritsa et al. [25] show that the marginal distribution of the cause is independent of the conditional distribution of the effect given the cause. To exploit this asymmetry, they propose the CURE algorithm, which is based on unsupervised reverse regression.

The algorithmic information-theoretic approach views causality in terms of Kolmogorov complexity. The key idea is that if X causes Y, the shortest description of the joint distribution $P(X, Y)$ is given by the separate descriptions of the distributions $P(X)$ and $P(Y \mid X)$ [11], and justifies additive noise model based causal inference [12]. However, as Kolmogorov complexity is not computable [15], causal inference using algorithmic information theory requires practical implementations, or notions of independence. For instance, the information-geometric approach [13] defines independence via orthogonality in information space for univariate continuous pairs. Vreeken [31] instantiates it with the cumulative entropy to infer the causal direction in continuous univariate and multivariate data. Mooij instantiates the first practical compression-based approach [18] using the Minimum Message Length. Budhathoki and Vreeken approximate $K(X)$ and $K(Y \mid X)$ through MDL, and propose ORIGO for causal inference on binary data [3]. Marx and Vreeken [16] propose SLOPE, an MDL based method employing local and global regression for univariate numeric data.

In contrast to all methods above, CRACK can consider pairs of any cardinality, univariate or multivariate, and of same, different, or even mixed-type data.

4 Causal Inference by Compression

We pursue the goal of causal inference by compression. Below we give a short introduction to the key concepts.

4.1 Causal Inference by Complexity

The problem we consider is to infer, given data over two correlated variables X and Y, whether X caused Y, whether Y caused X, or whether X and Y are only correlated. As is common in this setting, we assume causal sufficiency [17]. That is, we assume there exists no hidden confounding variable Z that causes both X and Y. This scenario is relevant not only when we are given only two variables and hence no conditional independence tests can be applied, but also when we are given a partially directed causal skeleton and orientation rules based on conditional independence tests can not resolve the remaining undirected edges.

The algorithmic Markov condition, as recently postulated by Janzing and Schölkopf [11], states that factorizing the joint distribution over *cause* and *effect* into $P(\textit{cause})$ and $P(\textit{effect} \mid \textit{cause})$, will lead to simpler—in terms of Kolmogorov complexity—models than factorizing it into $P(\textit{effect})$ and $P(\textit{cause} \mid \textit{effect})$. Formally, they postulate that if X causes Y,

$$K(P(X)) + K(P(Y \mid X)) \leq K(P(Y)) + K(P(X \mid Y)) \; .$$

While in general the symmetry of information, $K(x) + K(y \mid x) = K(y) + K(x \mid y)$, holds up to an additive constant [15], Janzing and Schölkopf [11] showed it does *not* hold when X causes Y, or vice versa. Hence, we can trivially define

$$\Delta_{X \to Y} = K(P(X)) + K(P(Y \mid X)) \; , \tag{1}$$

as a causal indicator that uses this asymmetry to infer that $X \to Y$ as the most likely causal direction if $\Delta_{X \to Y} < \Delta_{Y \to X}$, and vice versa.

This indicator assumes access to the true distribution $P(\cdot)$. In practice, we only have access to empirical data. Moreover, following from the halting problem, Kolmogorov complexity is not computable. We can approximate it, however, via MDL [15,7], which also allows us to directly work with empirical distributions.

4.2 Causal Inference by MDL

For causal inference by MDL, we will need to approximate both the marginals $K(P(X))$ and $K(P(Y))$ as well as the conditionals $K(P(Y \mid X))$ and $K(P(X \mid Y))$. For the former, we need to consider the model classes $\mathcal{M}_X$ and $\mathcal{M}_Y$, while for the latter we need to consider class $\mathcal{M}_{Y|X}$ of models $M_{Y|X}$ that describe the data of Y dependent the data of X, and accordingly for the inverse direction.

That is, we are after the *causal* model $M_{X \to Y} = (M_X, M_{Y|X})$ from the class $\mathcal{M}_{X \to Y} = \mathcal{M}_X \times \mathcal{M}_{Y|X}$ that best describes the data Y by exploiting as much structure of X as possible to save bits. By MDL, we identify the optimal model $M_{X \to Y} \in \mathcal{M}_{X \to Y}$ for data over X and Y as the one minimizing

$$L(X, Y, M_{X \to Y}) = L(X, M_X) + L(Y, M_{Y|X} \mid X) \; ,$$

where the encoded length of the data of X under a given model is encoded using two-part MDL, similarly so for Y, if we consider the inverse direction.

To identify the most likely causal direction between X and Y by MDL we can now simply rewrite Eq. (1) to define the *Absolute Causal Indicator* (ACI) [3]

$$ACI_{X \to Y} = L(X, M_X) + L(Y, M_{Y|X} \mid X) \; .$$

Akin to the Kolmogorov complexity based score, we infer that X is a likely cause of Y if $ACI_{X \to Y} < ACI_{Y \to X}$, Y is a likely cause of X if $ACI_{Y \to X} < ACI_{X \to Y}$.

4.3 Normalized Causal Indicator

The absolute causal indicator has nice theoretical properties that follow directly from the algorithmic Markov condition. However, by considering the absolute difference in encoded lengths between $X \to Y$ and $Y \to X$, it has an intrinsic bias towards data of higher marginal complexity. For example, when we gain 5 bits between encoding the data over Y conditioned on X, rather than independently, this is more impressive if $L(Y, M_Y)$ was 100 than when it was 1 000 000 bits. This is particularly important in the mixed-data case, as the marginal complexity of a binary attribute will typically be much smaller than that of a attribute recorded at a higher resolution.

To address this shortcoming in the ACI, we propose a novel, normalized indicator for causal inference on mixed-type data. We start with the ERGO indicator [31], which rather than the absolute difference considers the compression *ratios* of the target variables, i.e. iff $X \to Y$ then

$$\frac{L(X, M_{X|Y} \mid Y)}{L(X, M_X)} > \frac{L(Y, M_{Y|X} \mid X)}{L(Y, M_Y)} \; .$$

This score accounts for different marginal complexities of X and Y, and hence suffices for the univariate mixed-type data case. For the multivariate and mixed-type data case, we still face the same problem: if the variates of $Y_i \in Y$ are of different marginal complexities $L(Y_i, M_{Y_i})$, the gain in compression of one single Y_i may dominate the overall score simply because it has a larger marginal complexity than the others (e.g. because it has a larger domain).

We can compensate this by explicitly considering the compression ratios *per variate* $Y_i \in Y$, rather than the compression ratio over Y as a whole. Formally, we define our new *Normalized Causal Indicator* (NCI) as

$$NCI_{X \to Y} = \frac{1}{|Y|} \sum_{Y_i \in Y} \frac{L(Y_i, M_{Y_i|X} \mid X)}{L(Y_i, M_{Y_i})} \; .$$

As above, we infer $X \to Y$ if $NCI_{X \to Y} < NCI_{Y \to X}$ and vice versa.

Although free of bias from the marginal complexities of individual variates, we have to be careful to screen for redundancy within Y resp. X. By definition, the NCI counts the causal effect on each variate, and redundancies within Y (resp. X) hence exacerbate the measured effect. It is easy to detect redundancies within X resp. Y using standard independence tests, however.

In practice, we expect that ACI performs well on data where X and Y are of the same type, especially when $|X| = |Y|$ and the domain sizes of their attributes are balanced. Whenever the variates of X and Y are of different marginal complexities, e.g. because of unbalanced domains, dimensionality, and especially for mixed-type data, the experiments confirm that the NCI performs much better than the ACI.

5 MDL for Tree Models

To use MDL in practice, we need to specify an appropriate model class $\mathcal{M}$, and define how to encode both data and models in bits. Here, we need to be able to consider numeric, discrete and mixed-type data, be able to exploit dependencies between attributes of different types, and be able to encode the data of Y conditioned on the data of X. Classification and regression trees lend themselves very naturally to do all of this.

That is, we consider models M that contain a classification or regression tree T_i per attribute $A_i \in A$, where tree T_i encodes the data over A_i by exploiting dependencies on other attributes by means of splitting or regression. Together, the leaves of a tree encode the data of the attribute. Loosely speaking, the better we can fit the data in a leaf, the more succinctly we will be able to encode it.

A *valid* tree model M contains no cyclic dependencies between the trees $T_i \in M$, and hence a valid model can be represented by a DAG. Formally, we define $\mathcal{M}_A$ as the set of all valid tree models for data over a set of attributes A. We additionally define *conditional* tree models for Y given X, as the model class $\mathcal{M}_{Y|X}$ that consists of all valid tree models where we allow dependencies within Y, as well as from X to Y, but not from Y to X.

Cost of Data and Model Now that we know the relevant model classes, we can define our MDL score. At the highest level, the number of bits to describe data over attributes A together with a valid model M for A as

$$L(A, M) = \sum_{T_i \in M} L(A_i, T_i) \ ,$$

where we make use of the fact that M is a DAG, and we can hence serialize its dependencies.

In turn, the encoded cost of a tree T consists of two parts. First, we transmit its topology, and second the data in its leaves. For the topology, we indicate per node whether it is a leaf or an internal node, and if the latter, whether it is a split or regression node. Formally we hence have

$$L(A_i, T) = |T| + \sum_{v \in \text{int}(T)} (1 + L(v)) + \sum_{l \in \text{lvs}(T)} L(l) \ .$$

This leaves us to define the encoded cost of an internal node, $L(v)$, and the encoded cost of the data in the leaves, $L(l)$. We do this in turn.

Cost of a Node A node $v \in T$ can be of two main types; it either defines a split, or a regression step. We consider these in turn. We consider both multiway and single splits. To encode a split node v, we need

$$L_{\text{split}}(v) = 1 + \log |A| + L(\Phi_{\text{split}})$$

bits. We first encode whether it is a single or multiway split, then the attribute X_j, and last the conditions on which we split. For single way splits, $L(\Phi_{\text{split}})$ corresponds to the cost of describing the value in the domain of X_j on which we split, which is $\log |\mathcal{X}_j|$ when X_j is categorical, and $\log |\mathcal{X}_j| - 1|$ when it is binary or numeric. For multiway splits on categorical attributes X_j we split on all values, which costs no further bits, while for numeric X_j we split on every value that occurs at least k times—with one residual split for all remaining data points. To encode k we use $L_{\mathbb{N}}$, the MDL optimal code for integers [24].

To encode a regression node n, we first encode the attribute we regress on, and then the parameters $\Phi(v)$ of the regression, i.e.

$$L_{\text{reg}}(v) = \log |A| + \sum_{\phi \in \Phi(v)} \left(1 + L_{\mathbb{N}}(s) + L_{\mathbb{N}}(\lfloor \phi \cdot 10^s \rfloor) \right) .$$

We encode each parameter $\phi \in \Phi$ up to user defined precision, e.g. 0.001, by first encoding the corresponding number of significant digits s, e.g. 3, and then the shifted parameter value. In practice, for computational reasons, we use linear and quadratic regression, but note that this score is general for any regression technique with real-valued parameters.

Cost of a Leaf In classification and regression trees, the actual data is stored in the leaves. To encode the data in a leaf of a nominal attribute, we can use refined MDL [14]. That is, we are guaranteed to be as close to the number of bits we would need knowing the true model as possible, even if the true generating model is not in our model class [7]. In particular, we encode the data of a nominal leaf using the stochastic complexity for multinomials as

$$L_{\text{nom}}(l) = \log \sum_{h_1 + \cdots + h_k = |l|} \frac{|l|!}{h_1! h_2! \cdots h_k!} - |l| \cdot H(X_i \mid l) \ ,$$

where H denotes the Shannon entropy. Kontkanen & Myllymäki [14] derived a recursive formula to calculate this in linear time.

For numeric data, refined MDL encodings have very high computational complexity [14]. In the interest of efficiency, we hence encode the data in numeric leaves with two-part MDL, using point models with a Gaussian, resp. uniform distribution. The former is especially fitting after regression, since such a step aims to minimizes the variance of Gaussian distributed error. A split or a regression node can reduce the variance and, or the domain size of data in the leaf, and each can therewith reduce the cost. The costs for a numeric leaf are

$$L_{\text{num}}(l \mid \sigma, \mu) = \frac{|l|}{2} \left(\frac{1}{\ln 2} + \log 2\pi\sigma^2 \right) - |l| \log \text{res}(X_i),$$

given empirical mean μ and variance σ or as uniform given min and max as

$$L_{\text{num}}(l \mid \min(l), \max(l)) = |l| \cdot \log\left(\frac{\max(l) - \min(l)}{\text{res}(X_i)} + 1\right) .$$

We encode the data as Gaussian if this costs fewer bits than encoding it as uniform. To indicate this decision, we use one bit and encode the minimum of both plus the corresponding parameters. As we consider empirical data, we can safely assume that all parameters lie in the domain of the given attribute. The encoded costs of a numeric leaf l hence are

$$L_{\text{num}}(l) = 1 + 2\log|\mathcal{X}_i| + \min\{L_{\text{num}}(l \mid \sigma, \mu), L_{\text{num}}(l \mid \min(l), \max(l))\} .$$

We now have a complete score. In the next section we discuss how to optimize it, but first we discuss some important causal aspects.

Identifiability and Limitations Tree models are closely related to the algorithmic model of causality as postulated by Janzing and Schölkopf [11]. That is, every node X_i in a DAG can be computed by a program q_i with length $O(1)$ from its parents pa_i and additional input n_i—formally, $X_i = q_i(pa_i, n_i)$. Following the algorithmic Markov condition, the shortest description of X_i is through its parents.

In general, the MDL optimal tree model identifies the shortest description of a node A_i conditioned on a subset of attributes $S_i \subseteq A \backslash \{A_i\}$. In particular, by splitting or regressing on an attribute $A_j \in S_i$ it models program q_i given the parents as input. The remaining unexplained data that corresponds to the additional input or noise n_i is encoded in the leaves of the tree. In other words, tree T_i with the minimal costs relates to the tree where S_i contains only the parents of A_i, and encodes exactly the relevant dependencies towards A_i.

Although tree models are very general, we can identify specific settings in which the model is identifiable. First, consider the case where X and Y are univariate and of a single type. If both are numeric, our model reduces to a simple regression model, for which we know the correct causal direction can be identified based on regression error for non-linear function with Gaussian noise and linear functions with either data or noise not being Gaussian distributed [16,1]. Similarly, for discrete data we can identify additive noise models using stochastic complexity [2]. Since we model dependencies according to the algorithmic model of causality, we can generalize these concepts for multivariate data.

Further, it is easy to see that our score is monotone under subset restriction. That is, if $X \subseteq Y$, we can define $Z = Y \setminus X$, and have $L(Y) = L(X \cup Z) = L(X) + L(Z \mid X) \geq L(X)$. Additionally, it is submodular, $L(X \cup Z) - L(X) \geq L(Y \cup Z) - L(Y)$. As it also is trivially 0 for empty input, it is an information measure, and hence we know by the results of Steudel [28] that under our score tree models themselves are identifiable.

In practice, we are limited by the optimality of our approximation of the Kolmogorov complexity. That is, any inferences we make are with respect to

the encoding we defined above, rather than the much more generally defined Kolmogorov complexity. If the generating process does not use tree-models, or measures complexity differently, the inferences we draw based on our score may be wrong. The experiments show, however, that our scores are very reliable even in adversarial settings.

6 The Crack Algorithm

Finding the optimal decision tree for a single nominal attribute is NP-hard, and hence so is the optimization problem at hand. We introduce the CRACK algorithm, which stands for **c**lassification and **r**egression based p**ack**ing of data. CRACK is an efficient greedy heuristic for discovering a coding forest M from model class $\mathcal{M}$ for data over attributes A with low $L(A, M)$. It builds upon the well-known ID3 algorithm [22].

Greedy Algorithm We give the pseudocode of CRACK as Algorithm 1. Before running the algorithm, we set the resolution per attribute. To be robust to noise, we set $\text{res}(A_i)$ for continuous attributes to the k^{th} smallest distance between two adjacent values, with $k = 0.1 \cdot n$.

CRACK starts with an empty model consisting of only trivial trees, i.e. leaf nodes containing all records, per attribute (line 1). We iteratively discover that refinement of the current model that maximizes compression. To find the best refinement, we consider every attribute (4), and every legal additional split or regression of its corresponding tree (8). That is, a refinement is only legal when the dependency is allowed by the model family $\mathcal{M}$ (6–7) and the dependency graph remains acyclic.

The key subroutine of CRACK is REFINELEAF, in which we discover the optimal refinement of a leaf l in tree T_i. That is, it finds the optimal split of l over all candidate attributes A_j such that we minimize the encoded length. In case both A_i and A_j are numeric, REFINELEAF also considers the best linear and quadratic regression and decides for the variant with the best compression—choosing to split in case of a tie. In the interest of efficiency, we do not allow splitting or regressing multiple times on the same candidate.

Since we use a greedy heuristic to construct the coding trees, we have a worst case runtime of $O(2^m n)$, where m is the number of attributes and n is the number of rows. In practice, CRACK takes only a few seconds for all tested cause-effect pairs.

Causal Inference with Crack To compute our causal indicators we run CRACK twice on D. First with model class $\mathcal{M}_{X|Y}$ to obtain $M_{X|Y}$ and second with $\mathcal{M}_{Y|X}$, to obtain $M_{Y|X}$. For $L(X \mid M_X)$ we assume a uniform prior and define $L(X \mid M_X) = -n \sum_{A_i \in X} \log \text{res}(A_i)$ and do so analogue for Y. We refer to CRACK using NCI as CRACK_N, and as CRACK_A using ACI.

Algorithm 1: CRACK$(A, \mathcal{M})$

```
   input  : data over attributes A, model class M
   output : tree model M ∈ M with low L(A, M)
 1 T_i ← TrivialTree(A_i) for all A_i ∈ A;
 2 G ← (V = {v_i | i ∈ A}, E = ∅);
 3 while L(A, M) decreases do
 4     for A_i ∈ A do
 5         O_i ← T_i;
 6         for l ∈ lvs(T_i), (i, j) ∈ G do
 7             if E ∪ (v_i, v_j) is acyclic and j ∉ path(l) then
 8                 T'_i ← RefineLeaf(T_i, l, j);
 9                 if L(T'_i) < L(O_i) then
10                     O_i ← T'_i, e_i ← j;
11     k ← arg min_i{L(O_i) − L(T_i)};
12     if L(O_k) < L(T_k) then
13         T_k ← O_k, E ← E ∪ (v_k, v_{e_k}) ;
14 return M ← ⋃_i T_i
```

7 Experiments

In this section, we evaluate CRACK empirically. We implemented CRACK in C++, and provide the source code including the synthetic data generator along with the tested datasets for research purposes.[1] The experiments concerning CRACK were executed single-threaded on a MacBook Pro with 2.6 GHz Intel Core i7 processor and 16 GB memory running Mac OS X. All tested data sets could be processed within seconds; with a maximum runtime of 3.8 seconds.

7.1 Synthetic Data

On synthetic data, we want to show the advantages of either score. In particular, we expect CRACK$_A$ to perform well on nominal data and numeric data with balanced domain sizes and dimensions, whereas we expect CRACK$_N$ to perform better on numeric data with varying domain sizes and mixed-type data.

We generate synthetic data with assumed ground truth $X \to Y$ with $|X| = k$ and $|Y| = l$, each having $n = 5\,000$ rows, in the following way. First, we randomly assign the type for each attribute in X. For nominal data, we randomly draw the number of classes between two (binary) and five and distribute the classes uniformly. Numeric data is generated following a normal distribution taken to the power of q by keeping the sign, leading to a sub-Gaussian ($q < 1.0$) or super-Gaussian ($q > 1.0$) distribution.[2]

[1] `http://eda.mmci.uni-saarland.de/crack/`

[2] To ensure identifiability, we use super- and sub-Gaussians [9].

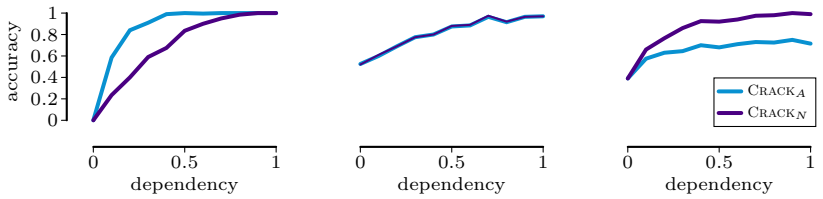

Fig. 1. Accuracy for *ACI* and *NCI* on nominal (left), numeric (middle) and mixed-type (right) data based on the dependency.

To create data with the true causal direction $X \to Y$, we introduce dependencies from X to Y, where we distinguish between splits and refinements. We call the probability threshold to create a dependency $\varphi \in [0, 1]$. For each $j \in \{1, \ldots, l\}$, we throw a biased coin based on φ for each $X_i \in X$ that determines if we model a dependency from X_i to Y_j. A split means that we find a category (nominal) or a split-point (numeric) on X_i to split Y_j into two groups, for which we model its distribution independently. As refinement, we either do a multiway split or model Y_j as a linear or quadratic function of X_i plus independent Gaussian noise.

Accuracy First, we compare the accuracies of CRACK$_N$ and CRACK$_A$ with regard to single-type and mixed-type data. To do so, we generate 200 synthetic data sets with $|X| = |Y| = 3$ for each dependency level where $\varphi \in \{0.0, 0.1, \ldots 1.0\}$. Figure 1 shows the results for numeric, nominal and mixed-type data. For single-type data, the accuracy of both methods increases with the dependency, and reaches nearly 100% for $\varphi = 1.0$. At $\varphi = 0$, both approaches correctly do not decide instead of taking wrong decisions. As expected CRACK$_N$ strongly outperforms CRACK$_A$ on mixed-type data, reaching near 100% accuracy, whereas CRACK$_A$ reaches only 72%. On nominal data, CRACK$_A$ picks up the correct signal faster than CRACK$_N$.

Dimensionality Next, we evaluate how sensitive both scores are w.r.t. the dimensionality of both X and Y, where we separately consider the cases of symmetric $k = l$ and asymmetric $k \neq l$ dimensionalities. Per setting, we consider the average accuracy over 200 independently generated data sets.

For the symmetric case, both methods are near to 100% on single-type data, whereas only CRACK$_N$ also reaches this target on mixed-type data, as can be seen in the appendix.[1] We now discuss the more interesting case for asymmetric pairs in detail. To test asymmetric pairs, we set the dimensionality of X to three, $|X| = 3$, and vary the dimensionality of Y from 1 to 11. To avoid bias, we choose the ground truth causal direction, i.e. $X \to Y$ and $Y \to X$, uniformly at random. We plot the results in Fig. 2. We observe that CRACK$_N$ has much less difficulty with the asymmetric data sets than CRACK$_A$. CRACK$_N$ performs near perfect and has a clear advantage over CRACK$_A$ on mixed-type and numeric data. CRACK$_A$ performs better on nominal only data for $l = 1$.

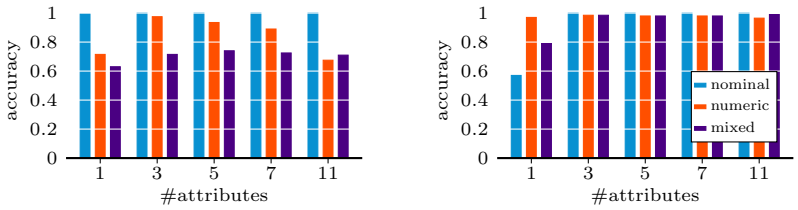

Fig. 2. Accuracy of *ACI* (left) and *NCI* (right) on for synthetically generated causal pairs of asymmetric cardinality, $|X| = 3$ and $|Y| \in \{1, 3, 5, 7, 11\}$ with ground truth $X \to Y$ or $Y \to X$ randomly chosen, for resp. nominal, numeric and mixed-type data.

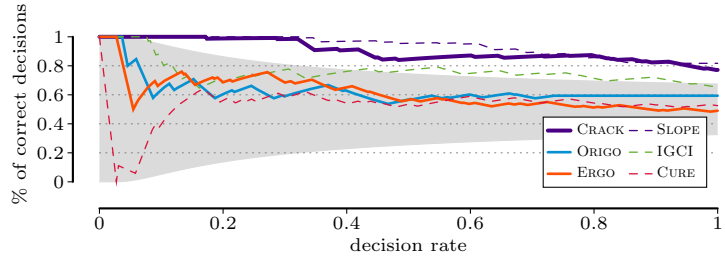

Fig. 3. [Higher is better] Decision rates of the multivariate methods CRACK, ORIGO and ERGO, and the univariate methods IGCI, CURE and SLOPE (dashed lines) on the univariate Tuebingen causal benchmark pairs (100), weighted as defined.

7.2 Univariate Benchmark Data

To evaluate CRACK on univariate data, we apply it to the well-known Tuebingen benchmark (v1.0) consisting of 100 univariate cause-effect pairs with known ground truth.[3] As these are mainly numeric pairs, with only a few categoric instances, we apply CRACK_A. We compare to the state of the art methods for multivariate pairs, ORIGO [3] and ERGO [31], and those specialized for univariate pairs, CURE [25], IGCI [13] and SLOPE [16] using their publicly available implementations and recommended parameter settings.

For each approach, we sort the results by confidence. Accordingly, we calculate the decision rate, the percentage of correct inferences up to each k inferences, weighting the decisions as specified by the benchmark. We plot the results in Fig. 3 and show the 95% confidence interval of a fair coin flip as a grey area. Except to CRACK none of the multivariate methods is significant w.r.t. the fair coin flip. In particular, CRACK has an accuracy of over 90% for the first 41% of its decisions and reaches 77.2% overall—the final result of CRACK_N is only 3% worse. CRACK also beats both CURE (52.5%) and IGCI (66.2%), which are methods specialized for univariate pairs. Perhaps most impressively, CRACK performs within the 95% confidence interval of the current state of the art on

[3] https://webdav.tuebingen.mpg.de/cause-effect/

Causal Pair	n	$\lvert X\rvert$	$\lvert Y\rvert$	Ground Truth	Type	Decisions			
						LTR	ERGO	ORIGO	CRACK
Climate	10 226	4	4	$Y \to X$	N	✓	✓	–	–
Ozone	989	1	3	$Y \to X$	N	(n/a)	✓	✓	✓
Car	392	3	2	$X \to Y$	N	–	✓	✓	✓
Radiation	72	16	16	$Y \to X$	N	–	–	–	✓
Symptoms	120	6	2	$X \to Y$	M	✓	✓	–	✓
Brightness	1 000	9	1	$X \to Y$	N	(n/a)	(n/a)	–	✓
Chemnitz	1 440	3	7	$X \to Y$	N	✓	✓	✓	✓
Precipitation	4 748	3	12	$X \to Y$	N	✓	–	–	✓
Stock 7	2 394	4	3	$X \to Y$	N	–	✓	–	✓
Stock 9	2 394	4	5	$X \to Y$	N	–	✓	–	✓
Haberman	306	3	1	$X \to Y$	M	✓	✓	–	–
Iris flower	150	4	1	$X \to Y$	M	(n/a)	(n/a)	–	✓
Canis	2 183	4	2	$X \to Y$	M	(n/a)	(n/a)	✓	✓
Lepus	2 183	4	3	$X \to Y$	M	(n/a)	(n/a)	✓	✓
Martes	2 183	4	2	$X \to Y$	M	(n/a)	(n/a)	✓	✓
Mammals	2 183	4	7	$X \to Y$	M	(n/a)	(n/a)	✓	✓
Octet	82	1	10	$Y \to X$	N	(n/a)	✓	✓	✓
Accuracy						0.56	0.82	0.47	0.88

Table 1. Comparison of LTR, ERGO, ORIGO and CRACK on 17 multivariate cause-effect pairs with known ground truth. The type is either "N" for numeric or "M" for mixed. A "✓" indicates a correct decision, a "–" an incorrect one and (n/a) that a method is not applicable.

causal inference on univariate numeric pairs, SLOPE, which has an overall accuracy of 81.7%. SLOPE is at the advantage for univariate pairs as it can exploit non-deterministic structure in the data. While interesting, this idea sadly does not seem to be efficiently applicable to multivariate data.

7.3 Real World Data

Next, we apply CRACK$_N$ on multivariate mixed-type and single-type data, where we collected 17 cause effect pairs with known ground truth. We provide basic statistics for each pair in Table 1. The first six are part of the Tuebingen benchmark [17], and the next four were provided by Janzing et al. [10]. Further, we extracted cause-effect pairs from the *Haberman* and *Iris* [5], *Mammals* [8] and *Octet* [6,29] data sets. More details are given in the appendix.

We compare CRACK$_N$ with LTR [10], ERGO [31] and ORIGO [3]. ERGO and LTR do not consider categoric data, and are hence not applicable on all data sets. In addition, LTR is only applicable to strictly multivariate data sets. CRACK$_N$ is applicable to all data sets, infers 15/17 causal directions correctly, by which it has an overall accuracy of 88.2%. Importantly, the two wrong decisions have low confidences compared to the correct inferences.

In addition, we conduct an experiment to check whether or not our result is influenced by redundant variables within X or Y. Hence, we first apply a standard redundancy test (R, Hmisc, redun) to omit redundant attributes within X or Y ($R^2 \geq 0.95$). After the reduction step, we apply CRACK$_N$ to the non-redundant pairs. As result, we found that the *Climate* cause effect pair indeed

contained redundant information and was inferred correctly after removing the redundant variables. For all other pairs, the prediction did not change. Hence, applying CRACK$_N$ after redundancy correction leads to an accuracy of 94.4%.

8 Conclusion

We considered the problem of inferring the causal direction from the joint distribution of two univariate or multivariate random variables X and Y consisting of single-, or mixed-type data. We point out weaknesses of known causal indicators and propose the normalized causal indicator for mixed-type data and data with highly unbalanced domains. Further, we propose a practical two-part MDL encoding based on classification and regression trees to instantiate the absolute and normalized causal indicators and provide CRACK, a fast greedy heuristic to efficiently approximate the optimal MDL score.

In the experiments, we evaluate the advantages of our proposed causal indicators and give advice on when to use them. On real world benchmark data, we are on par with the state of the art for univariate continuous data and beat the state of the art on multivariate data with a wide margin. For future work, we aim to investigate the application of CRACK for the discovery of causal networks as well as its application to biological networks.

Acknowledgements

The authors wish to thank Kailash Budhathoki for insightful discussions. Alexander Marx is supported by the International Max Planck Research School for Computer Science (IMPRS-CS). Both authors are supported by the Cluster of Excellence "Multimodal Computing and Interaction" within the Excellence Initiative of the German Federal Government.

References

1. P. Blöbaum, D. Janzing, T. Washio, S. Shimizu, and B. Schölkopf. Cause-effect inference by comparing regression errors. In *AISTATS*, 2018.
2. K. Budhathoki and J. Vreeken. MDL for Causal Inference on Discrete Data. In *ICDM*, pages 751–756, 2017.
3. K. Budhathoki and J. Vreeken. Origo: causal inference by compression. *Knowl. Inf. Sys.*, pages 1–23, 2017.
4. Z. Chen, K. Zhang, and L. Chan. Nonlinear causal discovery for high dimensional data: A kernelized trace method. In *ICDM*, pages 1003–1008. IEEE, 2013.
5. D. Dheeru and E. Karra Taniskidou. UCI machine learning repository, 2017.
6. L. M. Ghiringhelli, J. Vybiral, S. V. Levchenko, C. Draxl, and M. Scheffler. Big data of materials science: Critical role of the descriptor. *PRL*, 114, 2015.
7. P. Grünwald. *The Minimum Description Length Principle*. MIT Press, 2007.
8. H. Heikinheimo, M. Fortelius, J. Eronen, and H. Mannila. Biogeography of European land mammals shows environmentally distinct and spatially coherent clusters. *J. Biogeogr.*, 34:1053–1064, 2007.

9. P. Hoyer, D. Janzing, J. Mooij, J. Peters, and B. Schölkopf. Nonlinear causal discovery with additive noise models. In *NIPS*, pages 689–696, 2009.
10. D. Janzing, P. Hoyer, and B. Schölkopf. Telling cause from effect based on high-dimensional observations. In *ICML*, pages 479–486. JMLR, 2010.
11. D. Janzing and B. Schölkopf. Causal inference using the algorithmic markov condition. *IEEE TIT*, 56(10):5168–5194, 2010.
12. D. Janzing and B. Steudel. Justifying additive noise model-based causal discovery via algorithmic information theory. *OSID*, 17(2):189–212, 2010.
13. D. Janzing, J. Mooij, K. Zhang, J. Lemeire, J. Zscheischler, P. Daniušis, B. Steudel, and B. Schölkopf. Information-geometric approach to inferring causal directions. *AIJ*, 182-183:1–31, 2012.
14. P. Kontkanen and P. Myllymäki. MDL histogram density estimation. In *AISTATS*, pages 219–226, 2007.
15. M. Li and P. Vitányi. *An Introduction to Kolmogorov Complexity and its Applications*. Springer, 1993.
16. A. Marx and J. Vreeken. Telling Cause from Effect by MDL-based Local and Global Regression. In *ICDM*, pages 307–316. IEEE, 2017.
17. J. Mooij, J. Peters, D. Janzing, J. Zscheischler, and B. Schölkopf. Distinguishing cause from effect using observational data: Methods and benchmarks. *JMLR*, 17(32):1–102, 2016.
18. J. Mooij, O. Stegle, D. Janzing, K. Zhang, and B. Schölkopf. Probabilistic latent variable models for distinguishing between cause and effect. *NIPS*, 2010.
19. J. Pearl. *Causality: Models, Reasoning and Inference*. Cambridge University Press, 2009.
20. J. Peters, J. Mooij, D. Janzing, and B. Schölkopf. Causal discovery with continuous additive noise models. *JMLR*, 15:2009–2053, 2014.
21. J. Peters, D. Janzing, and B. Schölkopf. Causal Inference on Discrete Data using Additive Noise Models. *IEEE TPAMI*, 33(12):2436–2450, 2011.
22. J. R. Quinlan. Induction of decision trees. *Mach. Learn.*, 1(1):81–106, 1986.
23. J. Rissanen. Modeling by shortest data description. *Automatica*, 14(1):465–471, 1978.
24. J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals Stat.*, 11(2):416–431, 1983.
25. E. Sgouritsa, D. Janzing, P. Hennig, and B. Schölkopf. Inference of Cause and Effect with Unsupervised Inverse Regression. *AISTATS*, 38:847–855, 2015.
26. S. Shimizu, P. O. Hoyer, A. Hyvärinen, and A. Kerminen. A linear non-gaussian acyclic model for causal discovery. *JMLR*, 7:2003–2030, 2006.
27. P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search*. MIT press, 2000.
28. B. Steudel, D. Janzing, and B. Schölkopf. Causal markov condition for submodular information measures. In *COLT*, pages 464–476. OmniPress, 2010.
29. J. A. Van Vechten. Quantum dielectric theory of electronegativity in covalent systems. i. electronic dielectric constant. *PhysRev*, 182(3):891, 1969.
30. T. Verma and J. Pearl. Equivalence and synthesis of causal models. In *UAI*, pages 255–270, 1991.
31. J. Vreeken. Causal inference by direction of information. In *SDM*, pages 909–917. SIAM, 2015.
32. K. Zhang and A. Hyvärinen. On the identifiability of the post-nonlinear causal model. In *UAI*, pages 647–655, 2009.