Data-Driven Learning of Boolean Networks and Functions by Optimal Causation Entropy Principle (BoCSE)*

Jie Sun†, Abd AlRahman R AlMomani‡, and Erik Bollt§

Abstract. Boolean functions and networks are commonly used in the modeling and analysis of complex biological systems, and this paradigm is highly relevant in other important areas in data science and decision making, such as in the medical field and in the finance industry. In a Boolean model, the truth state of a variable is either 0 or 1 at a given time. Despite its apparent simplicity, Boolean networks are surprisingly relevant in many areas of application such as in bioinformatics to model gene expressions and interactions. In the latter case, a gene is either "on" or "off" depending on its expression level. Despite the promising utility of Boolean modeling, in most practical applications the Boolean network is not known. Automated learning of a Boolean network and Boolean functions, from data, is a challenging task due in part to the large number of unknowns (including both the structure of the network and the functions) to be estimated, for which a brute force approach would be exponentially complex. In this paper we develop a new information theoretic methodology that we show to be significantly more efficient than previous approaches. Building on the recently developed optimal causation entropy principle (oCSE), that we proved can correctly infer networks distinguishing between direct versus indirect connections, we develop here an efficient algorithm that furthermore infers a Boolean network (including both its structure and function) based on data observed from the evolving states at nodes. We call this new inference method, Boolean optimal causation entropy (BoCSE), which we will show that our method is both computationally efficient and also resilient to noise. Furthermore, it allows for selection of a set of features that best explains the process, a statement that can be described as a networked Boolean function reduced order model. We highlight our method to the feature selection in several real-world examples: (1) diagnosis of urinary diseases, (2) Cardiac SPECT diagnosis, (3) informative positions in the game Tic-Tac-Toe, and (4) risk causality analysis of loans in default status. Our proposed method is effective and efficient in all examples.

Key words. Boolean function, Boolean network, causal network inference, information flow, entropy, quantitative biology

AMS subject classifications. 62M86, 94A17, 37N99

1. Introduction.

In this paper we consider an important problem in data science and complex systems, that is the identification of the hidden structure and dynamics of a complex system from data. Our focus is on binary (Boolean) data, which commonly appears in many application domains. For example, in quantitative biology, Boolean data often comes from gene expression profiles where the observed state of a gene is classified or thresholded to either

*Submitted to the editors DATE.

Funding: This work was funded in part by the U.S. Army Research Office grant W911NF-16-1-0081 and the Simons Foundation grant 318812.

†Department of Mathematics, Clarkson University, Potsdam, NY 13699, and Clarkson Center for Complex Systems Science, Potsdam, NY 13699

‡Clarkson Center for Complex Systems Science, Potsdam, NY 13699, and Department of Electrical and Computer Engineering, Clarkson University, Potsdam, NY 13699

§Clarkson Center for Complex Systems Science, Potsdam, NY 13699, and Department of Electrical and Computer Engineering, Clarkson University, Potsdam, NY 13699

"on" (high expression level) or "off" (low to no expression level). In such an application, it is a central problem to understand the relation between different gene expressions, and how they might impact phenotypes such as occurrence of particular diseases. The interconnections among genes can be thought of as forming a Boolean network, via particular sets of Boolean functions that govern the dynamics of such a network. The use of Boolean networks has many applications, such as for the modeling of plant-pollinator dynamics [4], yeast cell cycles [17, 8], pharmacology networks [14], tuberculosis latency [12], regulation of bacteria [20], biochemical networks [13], immune interactions [29], signaling networks [32], gut microbiome [25], drug targeting [31], drug synergies [10], floral organ determination [2], gene interactions [1], and host-pathogen interactions [23]. In general, the problem of learning Boolean functions and Boolean networks from observational data in a complex system is an important problem to explain the switching relationships in these and many problems in science and engineering.

To date, many methods have been proposed to tackle the Boolean inference problem [18, 16, 9, 21, 30, 3, 19]. Notably, REVEAL (reverse engineering algorithm for inference of genetic network architectures), which was initially developed by Liang, Fuhrman and Somogyi in 1998 [18] has been extremely popular. REVEAL blends ideas from computational causality inference with information theory, and has been successfully applied in many different contexts. However, a main limitation of REVEAL is its combinatorial nature and thus suffers from high computational complexity cost, making it effectively infeasible for larger networks. The key challenge in Boolean inference are due to two main factors: (1) The system of interest is typically large, containing hundreds, if not thousands and more, components; (2) The amount of data available is generally not sufficient for straightforward reconstruction of the joint probability distribution. In this paper, we propose that information flow built upon causation entropy (CSE) for identifying direct versus indirect influences, [26, 27], using the optimal causation entropy principle (oCSE) [28] is well suited to develop a class of algorithms that furthermore enable computationally efficient and accurate reconstruction of Boolean networks and functions, despite noise and other sampling imperfections. Instead of relying on a combinatorial search, our method iteratively and greedily finds relevant causal nodes and edges and the best Boolean function that utilizes them, and thus is computationally efficient. We validate the effectiveness of our new approach that here we call, Boolean optimal causation entropy (BoCSE) using data from several real-world examples, including for the diagnosis of urinary diseases, Cardiac SPECT diagnosis, Tic-Tac-Toe, and risk causality analysis of loans in default status

This the paper is organized as follows. In Section 2 we review some basic concepts that define structure and function of Boolean networks and the problem of learning these from data. In Section 3 we present BoCSE as an information-theoretic approach together with a greedy search algorithm with agglomeration and rejection stages, for learning a Boolean network. In Section 4 we evaluate the proposed method on synthetic data as well as data from real-world examples, including those for automated diagnosis, game playing, and determination of causal factors in loan defaults. Finally, we conclude and discuss future work in Section 5, leaving more details on the basics of information theory in the Appendix.

2. The Problem of Learning a Boolean Network from Observational Data.

2.1. Boolean Function, Boolean Table, and Boolean Network. A function of the form

$$f : \mathbb{D} \to \mathbb{B}, \text{ where } \mathbb{B} = \{0, 1\}, \tag{1}$$

is called a Boolean function, where $\mathbb{D} \subset \mathbb{B}^k$ and $k \in \mathbb{N}$ is the arity of the function. For an k-ary Boolean function, there are 2^k possible input patterns, the output of each is either 0 or 1. The number of distinct k-ary Boolean functions is 2^{2^k}, a number that clearly becomes extremely large, extremely quickly, with respect to increasing k. Consider for example, $2^{2^3} = 256$, $2^{2^5} \approx 4.295 \times 10^9$, and $2^{2^8} \approx 1.158 \times 10^{77}$ (comparable to the number of atoms in the universe, which is estimated to be between 10^{78} and 10^{82}). This underlies the practical impossibility of approaching a Boolean function learning problem by brute force exhaustive search.

Each Boolean function can be represented by a truth table, called a *Boolean table*. The table identifies, for each input pattern, the output of the function. An example Boolean function $y = f(x_1, x_2, x_3)$ together with its corresponding Boolean table are shown in Fig. 1.

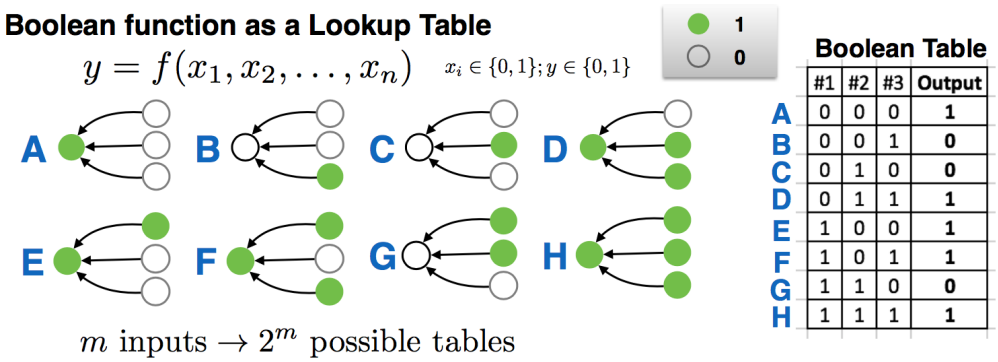

Figure 1. *A Boolean function can be uniquely identified by a truth table, called a Boolean table. For a k-ary Boolean function, the table has 2^k rows, and $k+1$ columns. For each row, the first k entries correspond to a particular input binary string (e.g., $(0, 1, 0)$), and the last entry represents the output of the function.*

A Boolean function is useful to model a system that has multiple binary inputs and a single binary output. More generally, a system can have multiple outputs, in which case multiple Boolean functions are needed, each capturing the relation between an output variable and the set of input variables. The collection of these individual functions constitute a *Boolean network*. Formally, a Boolean network is characterized by a triplet of sets, denoted as $G = (V; E; F)$, where $(V; E)$ represents a graph that encodes the structure of the network: $V(G) = \{1, 2, \ldots, n\}$ is the set of nodes, and $E(G) \subset V \times V$ is the set of directed edges (possibly including self-loops). The functional rules of the network are encoded in $F(G) = (f_1, f_2, \ldots, f_n)$, which is an ordered list of Boolean functions. For each node i in the network, we represent its set of *directed neighbors* by $\mathcal{N}_i = \{j : (i, j) \in E\}$ and the degree of node i as the cardinality of $\mathcal{N}_i$, denoted as $k_i = |\mathcal{N}_i|$. Thus, $f_i : \mathbb{B}^{k_i} \to \mathbb{B}$ is a k_i-ary Boolean function that represents the dependence of the state of node i on the state of its directed neighbors. Note that alternatively the dependence patterns of a Boolean network can also be

represented by an adjacency matrix $A = [A_{ij}]_{n\times n}$, where:

$$A_{ij} = \begin{cases} 1, & \text{if } j \in \mathcal{N}_i; \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Thus, the adjacency matrix A encodes the structure of a Boolean network, although not the functional rules.

2.2. Stochastic Boolean Function and Stochastic Boolean Network. In practice, the states and dynamics of a system are almost always subject to noise. Therefore, it is important to incorporate randomness and stochasticity into a Boolean network. To do so, we first extend the Boolean function concept from the deterministic definition to a stochastic generalization, defining a *stochastic Boolean function (SBF)* as

$$g(x) = f(x) \oplus \xi, \tag{3}$$

where f is a (deterministic) Boolean function and ξ is a Bernoulli random variable that controls the level of randomness of the function. In this model, the function contains a deterministic part, given by the Boolean function $f(x)$; the actual output of the function $g(x)$ is given by the output of $f(x)$ subject to a certain probably of being switched.

Following the notion of a stochastic Boolean function, we now define a *stochastic Boolean network* as a quadruple of sets, $G = (V; E; F; \boldsymbol{q})$, where the triplet of sets $(V; E; F)$ represents a (deterministic) Boolean network, and the vector $\boldsymbol{q} = [q_1, \ldots, q_n]^\top \in [0, 1]^n$ represents the level of noise, each as a random variable each with q_i quantifying the probability of switching the output state at node i, a scalar parameter describing the Bernoulli random variable $\xi_i \sim Bernoulli(q_i)$.

2.3. Data from Boolean Functions and Boolean Networks. We start by discussing several forms of data that commonly appear in application problems. These include: (a) Input-output data from a single Boolean function; (b) Input-output data from a Boolean network, which can be regarded as a generalization of (a); (c) Time series data from a Boolean network. In each one of these scenarios, the data can either be directly represented or rearranged into a set of input-output pairs

$$\{(\boldsymbol{x}(t), \boldsymbol{y}(t)) : t = 1, \ldots, T\}, \tag{4}$$

where,

$$\begin{aligned} &\boldsymbol{x}(t) = [x_1(t), \ldots, x_n(t)]^\top \in \mathcal{B}^n = \{0, 1\}^n, \text{ and,} \\ &\boldsymbol{y}(t) = [y_1(t), \ldots, y_\ell(t)]^\top \in \mathcal{B}^\ell = \{0, 1\}^\ell, \end{aligned} \tag{5}$$

are both vectors of Boolean states. We expand our discussion on this below.

(a) Input-output data from a single Boolean function. For a Boolean function (either deterministic or stochastic), if observations or measurements are made about its inputs and outputs, such data can be represented in the form of (4) where $\boldsymbol{y}(t)$ is a scalar (i.e., $\ell = 1$).

Here each pair $(\boldsymbol{x}(t), \boldsymbol{y}(t))$ represents the observed input string of k bits, encoded in $\boldsymbol{x}(t)$, and the corresponding output $\boldsymbol{y}(t)$. The ordering of the input-output pairs is arbitrary.

(b) Input-output data from a Boolean network. For a (deterministic or stochastic) Boolean network of n nodes, input-output data of the network comes in the form similar to that of a single Boolean function, except that each output itself it no longer a single bit, but instead multiple bits representing the state of all the nodes in the network. Thus, the dimensionality of $\boldsymbol{x}(t)$ and $\boldsymbol{y}(t)$ are both equal to n, that is, $\ell = n$ in the general form of (4). The ordering of input-output pairs is arbitrary.

(c) Time series data from a Boolean network. For a time series observed on a Boolean network of n nodes, we can represent such data using a sequence of Boolean vectors $(\boldsymbol{x}(t))_{t=0}^{T}$, where $\boldsymbol{x}(t) = [x_1(t), \ldots, x_n(t)]^\top \in \mathcal{B}^n$ represents the state of the entire network at time t. The pair $(\boldsymbol{x}(t-1), \boldsymbol{x}(t))$ can be described as an input-output data pair from the underlying Boolean network. For this matter, time series data from a Boolean network can also be put into the input-output data form (4) with the additional constraint that

$$\boldsymbol{y}(t) = \boldsymbol{x}(t+1) \text{ for every } t = 1, \ldots, T-1. \tag{6}$$

Here, unlike the case of input-output data as in (a) and (b), the temporal ordering in the time series data is unique and should not be (arbitrarily) changed.

To summarize, in these three commonly encountered scenarios as we discussed above, observational data from a Boolean network can be represented as input-output pairs as in (4). When the network contains only one node it is really just a Boolean function and thus each $\boldsymbol{y}$(t) is a scalar; on the other hand, when the data comes from time series then each $\boldsymbol{y}(t) = \boldsymbol{x}(t-1)$ and the temporal ordering of the data becomes fixed.

2.4. The Problem of Learning the Structure and Function of a Boolean Network.

Given Boolean data in the standardized form (4), we interpret data as samples of a multivariate conditional probability distribution

$$p(\boldsymbol{y}|\boldsymbol{x}) = \text{Prob}(Y(t) = \boldsymbol{y}|X(t) = \boldsymbol{x}) \tag{7}$$

$$= \prod_{i=1}^{k} \text{Prob}(Y_i(t) = y_i|X(t) = \boldsymbol{x}) = \prod_{i=1}^{k} p(y_i|\boldsymbol{x}), \tag{8}$$

where $\boldsymbol{x} \in \mathcal{B}^k$ and $\boldsymbol{y} \in \mathcal{B}^\ell$, and thus

$$p(y_i|\boldsymbol{x}) = p(y_i|x_1, \ldots, x_n). \tag{9}$$

The problem of reconstructing, or learning the Boolean network then is, can $p(y_i|\boldsymbol{x})$ be maximally reduced to a lower dimensional distribution. That is, does there exist a smallest (sub)set of indices,

$$S_i \subset \{1, \ldots, \ell\}, \text{ such that } p(y_i|\boldsymbol{x}) = p(y_i|\boldsymbol{x}_{S_i})? \tag{10}$$

Once we have identified, for each i, this set of nodes S_i, they together constitute a network, where a directed link $j \to i$ corresponds to having $j \in S_i$. Furthermore, to identify such a subset of explaining variables that closely approximates this conditional equality statement represents a simplified or reduced order presentation of the process.

3. BoCSE for Data-Driven Learning of the Structure and Function of Boolean Networks. In this section we develop a computational framework to reconstruct both the *structure* and *function* of a Boolean network from observational data. We start with the reconstruction of a *minimally sufficient* Boolean function from input-output data. This method is repeated to find the neighbor set and function for each node, and as a result reconstructs the whole network.

3.1. Reconstruction of a Minimally Sufficient Boolean function. Given a set of input-output pairs $\{\boldsymbol{x}(t), y(t)\}$, (here $y(t)$ is a single bit), we want to find a minimal Boolean function that is sufficient in representing the data. To quantify the complexity of the Boolean function, we state the following information-theoretic criterion

$$\begin{cases} \min_{K\subset[n]} |K|, \\ \text{s.t. } I(X^{(K)};Y) = \max_{K\in[n]} I(X^{(K)};Y) \end{cases} \tag{11}$$

Here,

$$\begin{aligned} [n &= \{1,2,\dots,n\}, \\ K &= \{k_1,\dots,k_\ell\} \text{ is a subset of } [n], \\ Y &= [y(1),\dots,y(t)]^\top, \text{ and} \\ X^{(K)} &= [X^{(K)}]_{T\times\ell} \text{ where } [X^{(K)}]_{tj} = X(t)_{k_j}. \end{aligned} \tag{12}$$

The symbol I denotes mutual information, that is, $I(X^{(K)};Y)$ is the mutual information between $X^{(K)}$ and Y.

At a glance, solving this combinatorial problem seems to be computationally complex. However, in our previous work [28] we developed an oCSE algorithm that can find K efficiently, and we proved in [28] that it correctly infers the underlying network as it is able to distinguish direct versus indirect connections correctly. Here we will further develop the concept to also learn the associated Boolean functions on the networks, that here we call BoCSE. Although various extensions of the oCSE algorithm are possible, some may even yield better results in certain scenarios. We focus here on the most basic version of our otherwise greedy search algorithm that consists of only two stages, a forward selection stage and a backward elimination stage.

- Forward selection. We initialize the solution set $K_f = \emptyset$, and, in each iteration, we choose an element k that satisfies the following conditions

$$\begin{cases} \max_j I(X_j;Y|X^{(K_f)}) > 0, \\ k = \arg\max_j I(X_j;Y|X^{(K_f)}). \end{cases} \tag{13}$$

  If such a k exists, then we append it to the set K_f and proceed to the next iteration; otherwise, when no such k exists, the forward selection is terminated.
- Backward elimination. Start with $K_b = K_f$, in each step of backward elimination, we select an element k that satisfy the following

$$k = \arg\min_{j\in K_b} I(X_j;Y|X^{(K_b/\{j\})}). \tag{14}$$

Such k always exists since K_b is a finite set. Then, if

$$I(X_k; Y | X^{(K_b/\{k\})}) = 0, \tag{15}$$

we remove k from K_b and repeat; otherwise, the algorithm terminates.
The result of the algorithm is a set $K_b = \{k_1, \ldots, k_\ell\}$, which is an estimate of the index set of the minimal Boolean function that fits data. Finally, given such a set K_b, we construct the corresponding Boolean function by estimating the best output (0 or 1) for each unique input pattern available from the data. Symbolically, for each $\boldsymbol{x}_0 \in \mathbb{B}^\ell$, we define the set

$$\mathcal{T}_{K_b}(\boldsymbol{x}_0) = \{t : \boldsymbol{x}^{(K_b)}(t) = \boldsymbol{x}_0\}, \tag{16}$$

and define

$$g(\boldsymbol{x}_0) = \frac{\sum_{t \in \mathcal{T}_{K_b}(\boldsymbol{x}_0)} y(t)}{|\mathcal{T}_{K_b}(\boldsymbol{x}_0)|} \in [0, 1]. \tag{17}$$

Then, we obtain $f : \mathbb{B}^\ell \to \mathbb{B}$ using the tabular form, by defining

$$f(\boldsymbol{x}_0) = \lceil g(\boldsymbol{x}_0) \rceil \in \{0, 1\}. \tag{18}$$

If $\mathcal{T}_{K_b}(\boldsymbol{x}_0) = \emptyset$ for some $\boldsymbol{x}_0$, it means that particular input pattern is never observed in the data. Then, in the absence of additional information, the value of f for such input cannot be optimally determined (the choice of either $f(\boldsymbol{x}_0) = 0$ or $f(\boldsymbol{x}_0) = 1$ makes no difference in "fitting" the data).

3.2. Estimation of Conditional Mutual Information and Tests of Significance. The proposed BoCSE learning approach requires estimating various forms of mutual information and conditional mutual information (see Appendix for their definition) from data. In practice (that is, when entropies need to be estimated from data), a threshold (either ε or η) needs to be determined in each step of either the forward or backward stage of the algorithm. The key is to decide, from data, whether an estimated conditional mutual information of the form $\hat{I} = I(X; Y|Z)$ should be regarded as zero, with confidence (as opposed to positive). In particular, we need to consider

$$\begin{cases} H_0(\text{null hypothesis}): \hat{I} = I(X; Y|Z) = 0, \\ H_1(\text{alternative hypothesis}): \quad \hat{I} = I(X; Y|Z) > 0. \end{cases} \tag{19}$$

To decide whether or not to reject H_0 (here equivalent as accepting H_1), we construct shuffled data by permuting the time ordering of the components in X. To be specific, suppose that

$$\sigma : \{1, \ldots, T\} \to \{1, \ldots, T\} \tag{20}$$

is a random permutation function, from which we compute $I(X^\sigma; Y|X_{\hat{S}})$ where X^σ represents the shuffled time series $\{x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(T)}\}$. By sampling σ uniformly, we then obtain a cdf

$$F(x) = P(I(X^\sigma; Y|X) \leq x). \tag{21}$$

From this cdf, we can then estimate the p-value under H_0 to be $1 - F(\hat{I})$, from which we can determine the threshold. For a given α-level (e.g., $\alpha = 0.01$), the corresponding threshold can then be decided as

$$\begin{cases} \varepsilon = F^{-1}(1-\alpha), & \text{for forward selection;} \\ \eta = F^{-1}(1-\alpha), & \text{for backward elimination.} \end{cases} \tag{22}$$

Throughout this paper, we set the same $\alpha = 0.05$ for both the forward and backward stage of the algorithm (unless otherwise noted), and obtain the cdf $F(x)$ by uniformly sampling by selecting 1000 independent random permutation functions σ.

4. Examples of Applications.

In this section we now present examples of applications of BoCSE, the proposed Boolean learning method.

4.1. Benchmark on Random Boolean Networks.

We first evaluate BoCSE for learning randomly generated Boolean networks. These networks are generated with two parameters, n is the number of nodes, and K is the in-degree of each node in the network (for example, $K = 3$ means that each node i receives three inputs from other nodes, randomly chosen). The Boolean function associated with each node i is constructed by assigning randomly an output of either 0 or 1 to each input pattern, with the equal probability. Figure 2 shows that, the number of data points needed for correctly learning the entire Boolean network scales sublinearly as the size of the network (left panel). Although the scaling becomes worse as K increases, it is still within practical reach for networks of several hundred of nodes. In the right panel of Fig. 2, we show the error of learning for networks of fixed size $n = 50$. As the length of time series increases (more data points), both false positive and false negative ratios decrease toward zero, confirming the validity and convergence of the method.

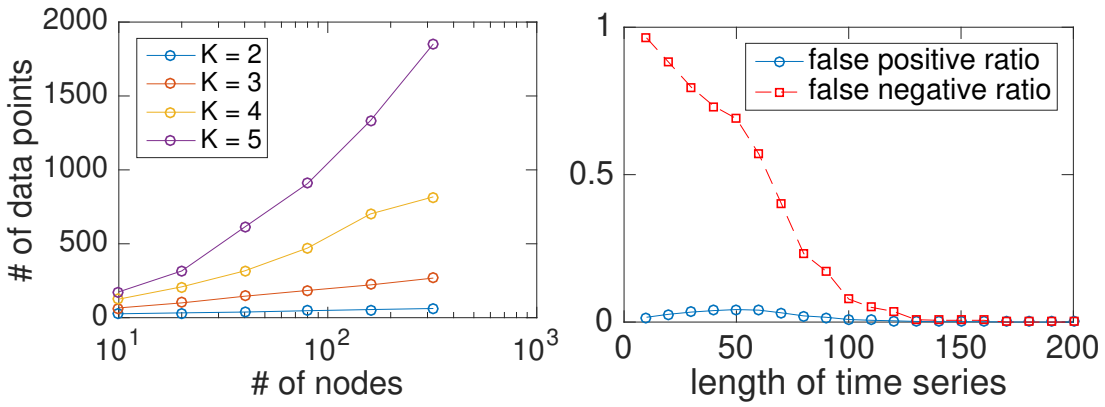

Figure 2. *(Left) Number of data points required for learning random Boolean networks with no error. (Right) Error ratios as a result of applying the proposed method BoCSE for learning random Boolean networks of fixed size $n = 50$ and degree $K = 3$. For both panels, each point on the plotted curve is the result of an average over 50 random realizations.*

4.2. Automated Diagnosis of Urinary Diseases.

As an application to aid the automation of medical diagnosis, we consider a dataset that documents the symptoms and diagnosis

outcomes of 120 patients. The data is an extended version of the table used in Ref. [7], and is available at the UCI Machine Learning database, via the following link under the name "Acute Inflammations": https://archive.ics.uci.edu/ml/datasets/Acute+Inflammations.

In this extended dataset, there are descriptions from a total of 120 patients, each with 6 attributes and 2 decision variables. The attributes are: (1) temperature, (2) nausea, (3) lumbar pain, (4) urine pushing, (5) micturition pains, (6) burning of urethra, itch, swelling of urethra outlet. Other than temperature, which takes value in the range of 35-42°C), all the other 5 attributes are recorded as a Boolean value, either "1" (symptom exists) or "0". In our analysis, we threshold the temperature data into binary values by simple thresholding: temperature equal or above 38°C are converted into "1" (fever) and those below are converted into "0" (no fever). The two decision (outcome) Boolean variables are

1. (acute) inflammation of urinary bladder,
2. nephritis of renal pelvis origin.

In Table 1 we summarize the description of the attributes and decision variables.

attributes	description
X_1	fever
X_2	nausea
X_3	lumbar pain
X_4	urine pushing
X_5	micturition pains
X_6	burning of urethra, itch, swelling of urethra outlet

outcome	description
Y_1	(acute) inflammation of urinary bladder
Y_2	nephritis of renal pelvis origin

Table 1

Attributes and outcome variables for the urinary disease data. Each variable is Boolean and takes value 1 *or* 0 *representing the presence or absence of a particular attribute/outcome.*

We apply the BoCSE learning method separately to the two outcome variables. For each outcome variable, we treat each patient's attributes as one input Boolean string and the corresponding recorded outcome as a single output.

For the first outcome variable Y_1, that is the inflammation of urinary bladder, we found that the relevant attributes are (in terms of decreasing order of importance): (4) urine pushing, (5) micturition pains, and (6) burning of urethra, itch, swelling of urethra outlet. The inferred Boolean function for the relation between these attributes and the outcome are shown in the left part of Table 2, and is found to accurately describe every individual data record. Interestingly, for the other outcome Y_2, the relevant attributes become X_1 and X_3 (in the order of decreasing importance), and the inferred Boolean function as shown in the right table of Table 2, can be written using a simple "and" gate: $Y_2 = X_1 \wedge X_2$, which implies that the diagonals of nephritis of renal pelvis origin can be based on having both symptoms: fever and lumbar pain. Yet again, this relation is consistent with every single patient's record in the dataset.

Next, using the inferred attributes from the entire dataset (120 samples), we explore the dependence of the accuracy of our Boolean inference on the sample size. We do this by

X_4	X_5	X_6	Y_1	Occurrence
0	0	0	0	25.00%
0	0	1	N/A	0%
0	1	0	0	8.33%
0	1	1	N/A	0%
1	0	0	1	8.33%
1	0	1	0	17.50%
1	1	0	1	16.67%
1	1	1	1	24.17%

X_1	X_3	Y_2	Occurrence
0	0	0	33.33%
0	1	0	16.67%
1	0	0	8.33%
1	1	1	41.67%

Table 2

Inferred Boolean relations by BoCSE for the two outcome variables: Y_1 (left table) and Y_2 (right table). Each entry in the "occurrence" column shows the fraction of observed attribute data: (X_4, X_5, X_6) for the left table and (X_1, X_3) for the right table. For each attribute pattern, the "predicted" value of outcome is shown in the Y column, where "N/A" refers to cases where no such input pattern is ever seen in the empirical data.

randomly selecting a subset of the samples, and use such "down-sampled" data instead of the full dataset for Boolean inference. For each sample size, we repeat such inference 50 times and compute the average number of false positives (attributes inferred using the down-sampled data that are not present using the full-size data) and false negatives (attributed to inferrence using the full data set which now appears using the down-sampled data). The results are shown in Fig. 3. Interestingly, for this particular example our method never seems to produce false positives, and the number of false negatives decrease rapidly to zero as more samples are used in Boolean inference, which suggests effectiveness of the method in automated diagnosis systems via relatively small sample sizes.

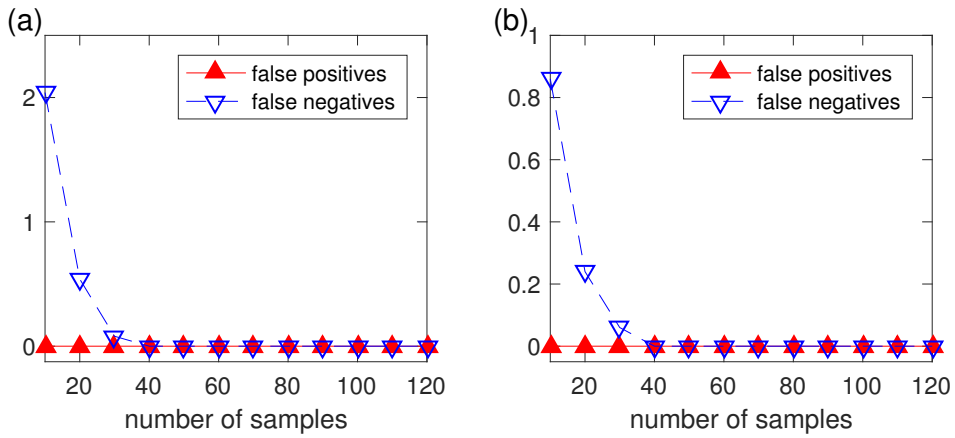

Figure 3. *Boolean inference error as a function of sample size for the urinary disease example. Here the "true" set of relevant attributes are taken to be the ones inferred using the full data (120 samples). (a) Inference error as a function of sample size (from 10 to 120) for the first outcome variable, Y_1, where error is quantified by false positives and false negatives. The number of true attributes is 3 in this case. (b) Similar to (a), but for Y_2. The number of true attributes is 2 in this case. In each plot, each data point is an average over 50 independent random down-sampling of the full dataset.*

4.3. Automated Cardiac SPECT Diagnosis. In this example, we test our Boolean learning method on an existing dataset that aims at automated image-based cardiac diagnosis. The dataset is derived from a set of images obtained by cardiac Single Proton Emission Computed Tomography (SPECT) [15]. In particular, there is a total of $T = 267$ patients, each of whom is classified as either normal ($y_t = 1$) or abnormal ($y_t = 0$). The data is divided into a training set which contains $T_1 = 80$ patients and a test (validation) set of $T_2 = 187$ patients. For each patient's image set, a total of $n = 22$ binary feature patterns were created, defining $x_i(t) = 1$ if the i-th feature is present in the SPECT images of the t-th patient, and $x_i(t) = 0$ otherwise, for $i = 1, \ldots, 22$ and $t = 1, \ldots, 267$. Finally, this post-processed Boolean dataset is further divided into a training set which contains 87 out of the 267 patients' features and diagnosis, and a validation set which contains such information for the remaining 180 patients.

Focusing on this post-processed Boolean data, we are interested to see if our automated Boolean inference method is able to learn the decision rules, that is, to diagnose a patient based on a reduced set of Boolean features out of the 22 features. In this sense, our methodology can be understood as useful for reduced order modelling (ROM) in the realm of complex Boolean function inference problems. Said another way, this method describes a way to simplify decision making problems by focusing on the most relevant factors that are those that lead to important outcomes. As shown in Fig. 4, our method is able to learn a Boolean function that achieves near 80% of decision accuracy on the validation data across a wide range of parameters. The our achieved accuracy, generally using only a subset of the full set of 22 Boolean features, and without any fine-tuning of parameters or further optimization, is already comparable to the best known result on such datasets [5].

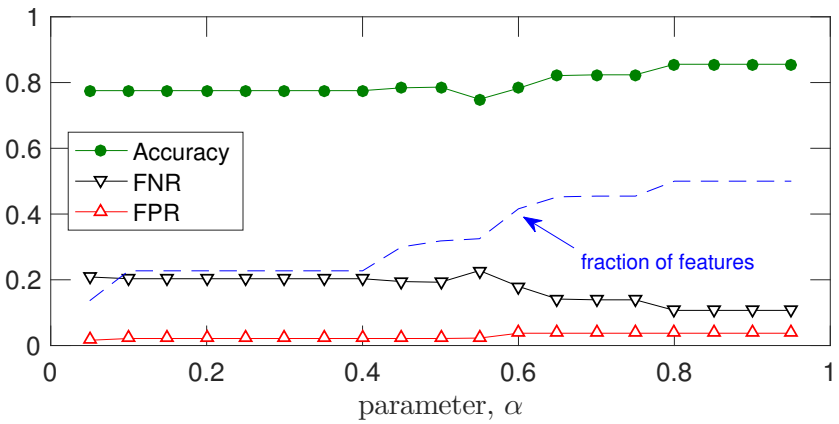

Figure 4. *Automated diagnosis of heart disease using 22 Boolean attributes derived from cardiac SPECT. Here we explore how the diagnosis accuracy changes as the parameter α in our Boolean inference method is varied. In particular, we apply BoCSE to the training data (80 patients) and validate the resulting Boolean functions on the validation set (187 patients). We compute the accuracy of diagnosis as the overall percentage of correct diagnosis in the validation set, shown in the figure. In addition, we also compute and plot, for each α, false positive ration (FPR) and false negative ratio (FNR), together with the effective number of Boolean variables inferred by our method (dashed curve).*

4.4. Tic-Tac-Toe. The Tic-Tac-Toe is a classical two-player board game, which is also often played on pencil-and-paper. The "board" is a 3-by-3 grid with a total of 9 slots, as

illustrated in Fig. 5(a). At the beginning of the game, the board is empty. Then, the two players take turn to mark any empty "slot" in each turn—typically one uses "X" the other uses "O". The player that is the first to have marked three consecutive horizontal, or vertical, or diagonals slots, wins the game. For instance, Fig. 5(b-d) is an example of the sequence of marks made by the players, where the first player (player "X") eventually wins by having marked an entire row (in this case, the top row). In general, if both players do their best at every move, the outcome would be a draw.

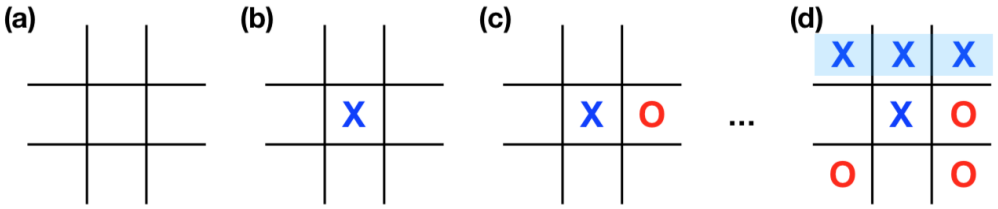

Figure 5. *Tic-Tac-Toe game. (a) Start of the game, where the board is made up of a 3-by-3 grid. (b-d) Example of a sequence of moves made by two players, where player "X" plays first, and eventually won the game by filling in an entire horizontal row.*

Our interest here is not (re)analysis strategies of this relatively simple game. For those who are interested, variants of the game actually has a connection to Ramsey theory, [11, 22]. Here, we are intersted in testing our Boolean learning algorithm to see if it provides any useful information. To this end, we collected the complete set of possible board configurations at the end of a tic-tac-toe game, via the following link under the name "Tic-Tac-Toe Endgame Data Set ": https://archive.ics.uci.edu/ml/datasets/Tic-Tac-Toe+Endgame. There is a total of 958 instances. For the t-th instance, we use $\boldsymbol{x}(t) = [x_1(t), \ldots, x_9(t)]^\top$ to present the state of the i-th slot, ordered as follows: upper-left, upper-middle, upper-right, middle-left, center, middle-right, lower-left, lower-middle, and lower-right. Each $x_i(t)$ can either be 1 (if marked by "X"), -1 (if marked by ("O"), or 0 (if empty). Corresponding to each instance t is the final outcome, which we denote as $y(t)$, which either equals 1 if player "X" wins or 0 if "X" does not win. Interpreting (t) and $y(t)$ as samples of random variables $\boldsymbol{X}$ and $\boldsymbol{Y}$, we can ask the question of which slots in the board, statistically, are more relevant (or predicative) for the first player to win the game.

Applying our Boolean learning algorithm, we found a list of most important slots, ordered in decreasing value of (added) relevance: $i_1 = 5$ (the center), $i_2 = 1$ (upper-left), $i_3 = 9$ (lower-right), $i_4 = 3$ (upper-right), $i_5 = 7$ (lower-left), $i_6 = 8$ (lower-middle), and $i_7 = 2$ (lower-right). To quantify the relative importance of each attribute, we compute the conditional entropy $H(Y|X_{i_1 \ldots i_k})$ for $k = 0, \ldots, 7$, where $H(Y|X_{i_0})$ is used to represent $H(Y)$. This shows that, as the number of attributes increases, uncertainty decreases monotonically and reaches 0 (complete predictability) when 7 attributes are used.

4.5. Risk Causality Analysis of Loans in Default Status. Loan default prediction is an essential problem for the banks and insurance companies to fiscally responsibly approve loans. However, in many cases, the borrowers fail to pay the loan as agreed, called loan default, which motivates the risk analysis problem in the banking industry, to identify those parameters that

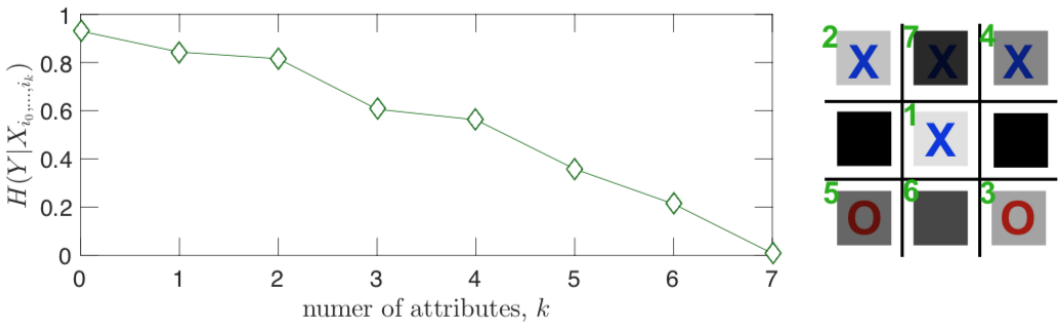

Figure 6. *Decrease of uncertainty in "predicting" the outcome of a Tic-Tac-Toe game using partial observations of the board in the final configuration. Here uncertainty is measured by the conditional entropy $H(Y|X_{i_1\ldots i_k})$, and the indices i_k are obtained using our Boolean inference algorithm: $i_1 = 5$ (the center), $i_2 = 1$ (upper-left), $i_3 = 9$ (lower-right), $i_4 = 3$ (upper-right), $i_5 = 7$ (lower-left), $i_6 = 8$ (lower-middle), and $i_7 = 2$ (lower-right).*

identify one borrower as trustworthy, and another borrower as representing a high risk.

We consider the open dataset from LendingClub (American peer-to-peer lending company), which can be downloaded from LendingClub website. We considered the dataset for the year 2019 (four quarters). The dataset contains more than 500,000 entries (data points, sample size). However, we only considered the long term (the final) status of the loans. Therefore, we excluded all the loans with the status *"Current"* as an outcome, to have a sample size of 62,460 for our analysis. That is, all can be classified to and outcome *"Paid in full"*, or *"Default"* status. We should emphasize here that we only considered the parameters as Boolean in nature, which limits the considered to those 10 parameters that we investigate as to their influence on the outcomes.

This example gives a causality driven description of those parameters that *combined*=, can represent a high risk that the borrower will not be able to pay his loan in full. This causality inference occurs within the Boolean framework for parameters concerning the loan long term status. Table 3 shows the attributes and their description. In loan issuing risk analysis, the amount of the requested loan and the annual income of the borrower are important variables to consider, and they are both numeric variables. We introduce here the combined attribute, loan to income ratio, which combines both variables in the form of a Boolean variable. Our dataset has a sample size of 62,460, and the loan to income ratio range from 0.0001 to 36000. So, we considered the median value $\mu \approx 0.2$ to be our threshold step, such that $X_9 = 0$, indicate that the loan to income ratio of the loan request is less than 0.2, and it is within the lowest 50% of all the requested loans over the period (which is in our case, one year).

We then apply BoCSE, the proposed Boolean factors learning approach to the outcome variable Y, where we found that the relevant attributes are, in decreasing order of importance:

- X_9, Loan to income ratio.
- X_{10}, Loan terms.
- X_3, Verification of the reported income.

We expect that the probability of having the loan fully paid ($Y = 1$) will be larger than a default ($Y = 0$) in this example, for all the observed combinations of the relevant attributes. However, the challenge here is to find the combination of attributes that, together represent a high risk if approving the loan. For example, if for some attribute combinations (binary string) X, the probability $Pr(Y = 1) = 0.8$, and $Pr(Y = 0) = 0.2$, we may not be satisfied by saying that the expected outcome that the borrower will pay the loan in full because that $Pr(Y = 1) > Pr(Y = 0)$. Our focus here will be that there is a risk that the borrower will not pay the loan with probability $Pr(Y = 0) = 0.2$, which represents a high risk.

In Table 4, we can see the inferred Boolean function relationship between these attributes and the resulting outcomes. From application of our automated Boolean function learning method, result shown in Table 4 we summarize these interesting summary observations:

- The first four rows represent patterns where $X_3 = 0$, that describe loans in which the borrower's income was not verified. We see that this feature coincides with a significant increase in the probability that the loan will not be paid in full. The lowest value in this group of patterns is $(X_3, X_9, X_{10}) = (0, 0, 1)$, which represents an unverified income, low loan to income ratio, and 60 months loan term, and the joint probability is then 1.9%. We conclude that low loan to income ratio combined with long term loan (which implies low monthly payment) reduces the risk of loan default.
- For the same pattern, but with a 36 months loan term, $(X_3, X_9, X_{10}) = (0, 0, 0)$, we see a significant increase in risk from 1.9% to 9.6%. For the pattern $(X_3, X_9) = (0, 1)$,the risk increases with the 36-month term loan, from 6.3% to 8.1%. This indicates that higher monthly payments indicate a higher risk. However, the effect of the loan term becomes neutral, meaning no effect in terms of observing only the verified income patterns ($X_3 = 1$). For these two patterns, where we have a verified income, the same risk conclusions follow for both the 36 and 60 months terms loans.
- Comparing the above two points, we conclude that if the borrower's reported income is verified, there is no difference in the risk between different loan terms. However, if it is not verified, then going with the 60 months term loans can profoundly reduce the risk, regardless of the loan to income ratio.
- We see that the lowest risk, or the most trustworthy borrowers, are the ones with the combination $(X_3, X_9, X_{10}) = (1, 0, 1)$, which represent a verified income, low loan to income ratio, and 60 months term loan reflecting low monthly payments. The risk, in this case, is about 0.4%. Unfortunately, however, such a pattern occurs infrequently, representing fewer than 1% of the borrower customers.
- On the other hand, we see that a significant high risk associates with the combination $(X_3, X_9, X_{10}) = (0, 0, 0)$, which represents unverified income, low LTI ratio, and 36 months term loan (high monthly payments). This is particularly interesting since a low LTI may on its own may suggest a safer primary indication because it implies a high income, low loan value, or both. However, we see that even with high income, or low requested loan amount, the combination of unverified income together with large monthly payment, $(X_3, X_{10}) = (0, 0)$, has the highest risk as compared to all other combinations in the table, 9.6% and 8.1%.

attributes	description
X_1	Home ownership. (1: Homeowner, 0: Not a homeowner)
X_2	Delinquency in the past two years. (1: Delinquency occurred, 0: No Delinquency)
X_3	Verification of the reported income. (1: Income verified, 0: Income not verified)
X_4	History of public records. (1: There is a public record, 0: No public records.)
X_5	Application type. (1: Individual, 0: With co-borrower)
X_6	120 days past due. (1: Have account that ever past due more than 120 days, 0: Never past due more than 120 days.)
X_7	Recent opened accounts in the last 12 months. (1: Have opened a new account in the last 12 months, 0: No new accounts)
X_8	Bankruptcies. (1: have declared bankruptcies, 0: Never declared bankruptcies)
X_9	Loan to income ratio. See caption.
X_{10}	Terms. (1: 60 months term, 0: 36 months term)

Table 3

Attributes for the loan issuance data. All variables are Boolean. The outcome Y is a Boolean vector that take the value 1 if the loan fully paid, and 0 otherwise (charged off or marked as default). The loan to income ratio is the ratio $r = \frac{\text{the loan amount}}{\text{annual income}}$, and it is formed as a Boolean function such that $X_9 = \begin{cases} 1, r > \mu \\ 0, r \leq \mu \end{cases}$, where μ is a threshold ratio that we selected to be the median value of the ratio of all the available dataset, and it was $\mu = 0.2$.

5. Discussion and Conclusion. Although black-box machine learning methods become increasingly more popular due to their relative ease to implement without deep understanding of how they work, in some applications such as quantitative biology where it is essential to uncover causal and relevant factors beyond functional fitting. A prototype problem of such is to learn, from noisy observational data, the structure and function of a Boolean network. The classic widely used REVEAL approach accomplishes this by performing a combinatorial search in the space of Boolean variables, and its performance relies heavily on having a relatively small network size and small maximum degree, two aspects that are in sharp contrast to typical biological systems that can be large and complex. To overcome these difficulties, here we present BoCSE as a new learning approach based on the optimization of causation entropy applying to Boolean data. This new approach relies on computing entropies of judiciously constructed subsets of variables, and does not require the combinatorial search typically used in other methods. We benchmark effectiveness of BoCSE on random Boolean networks, and further apply it in several real-world datasets, including in finding the minimal relevant diagnosis signals, quantifying the informative signs of a board game Tic-Tac-Toe, and in determining the causal signatures in loan defaults. In all examples, the BoCSE provides outcomes that is directly interpretable and relevant to the respective application scenarios.

X_3	X_9	X_{10}	Occurrence P_o	$Pr(Y=0\|X)$	$Pr(Y=0,X)$
0	0	0	38.75%	24.83%	9.6%
0	0	1	5.93%	33.66%	1.9%
0	1	0	23.29%	34.83%	8.1%
0	1	1	15.80%	39.92%	6.3%
1	0	0	4.35%	31.72%	1.4%
1	0	1	0.98%	38.59%	0.4%
1	1	0	5.78%	41.46%	2.4%
1	1	1	5.13%	46.28%	2.4%

Table 4

Inferred Boolean relations for the outcome variable Y. Each entry in the "occurrence" column shows the fraction of observed attribute data (X_3, X_9, X_{10}). Given each attribute pattern, the conditional probability that value of outcome $Pr(Y=0|X=(x_3,x_9,x_{10}))$, meaning that the probability that the borrower will not fully pay the loan, is shown in the $Pr(Y=0)$ column. The most important quantity that should be consider in the analysis is the joint probability for the pattern occurrence and the outcome $Y=0$, $Pr(Y=0, X=(x_3,x_9,x_{10}))$ is shown in last column of the table.

Appendix A. Basic concepts from information theory. In this appendix we review some basic concepts from information theory. These concepts are rooted in information and theory [24, 6], and are heavily utilized in our computational approach for Boolean inference.

The (Shannon) *entropy* of a discrete random variable X is given by

$$H(X) = -\sum_x P(x)\log P(x), \tag{23}$$

where $P(x) = \text{Prob}(X = x)$ and the summation is over the support of $P(x)$, that is, all values of x for which $P(x) > 0$. The base of the "log" function is typically chosen to be 2 so that the unit of entropy becomes "bit", although other base values can also be used depending on the application. Entropy is a measure of "uncertainty" associated with the random variable: generally the larger the entropy is the more difficult it is to "guess" the outcome of a random sample of the variable.

When two random variables X and Y are considered, we denote their joint distribution by $P(x,y) = \text{Prob}(X = x, Y = y)$ and conditional distributions by $P(y|x) = \text{Prob}(Y = y|X = x)$ and $P(x|y) = \text{Prob}(X = x|Y = y)$, respectively. These functions are used to define the *joint entropy* as well as the *conditional entropies*, as:

$$\begin{cases} \text{Joint entropy: } H(X,Y) = -\sum_{x,y} P(x,y)\log P(x,y), \\ \text{Conditional entropies:} \\ \quad Y \text{ given } X\text{: } H(Y|X) = -\sum_{x,y} P(x,y)\log P(y|x), \\ \quad X \text{ given } Y\text{: } H(X|Y) = -\sum_{x,y} P(x,y)\log P(x|y). \end{cases} \tag{24}$$

While the joint entropy $H(X,Y)$ measures the uncertainty associated with the joint variable (X,Y), the conditional entropy $H(Y|X)$ measures the uncertainty of Y given knowledge about X and similar interpretation holds for $H(X|Y)$. In general, $H(Y|X) \leq H(Y)$ and

$H(X|Y) \leq H(X)$, with "=" if and only if X and Y are independent. Interestingly, the reduction of uncertainty as measured by $H(Y) - H(Y|X)$ coincides with $H(X) - H(X|Y)$, leading to a quantity called the the *mutual information* (MI) between X and Y, given by:

$$I(X;Y) = H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{25}$$

Mutual information is symmetric $I(X;Y) = I(Y;X)$, and also nonnegative: $I(X;Y) \geq 0$ with $I(X;Y) = 0$ if and only if X and Y are independent.

Finally, the *conditional mutual information* (CMI) between X and Y given Z is

$$I(X;Y|Z) = H(X|Z) - H(X|Y,Z), \tag{26}$$

which measures the reduction of uncertainty of X given Z due to extra information provided by Y. Conditional mutual information is symmetric with respect to interchanging X and Y, and nonnegative, equalling zero if and only if the conditional probabilities $P(x|z)$ and $P(y|z)$ are independent: $P(x|z)P(y|z) = P(x,y|z)$.

REFERENCES

[1] M. C. Álvarez-Silva, S. Yepes, M. M. Torres, and A. F. G. Barrios, *Proteins interaction network and modeling of igvh mutational status in chronic lymphocytic leukemia*, Theoretical Biology and Medical Modelling, 12 (2015), p. 12.

[2] E. Azpeitia, J. Davila-Velderrain, C. Villarreal, and E. R. Alvarez-Buylla, *Gene regulatory network models for floral organ determination*, in Flower Development, Springer, 2014, pp. 441–469.

[3] N. Berestovsky and L. Nakhleh, *An evaluation of methods for inferring boolean networks from time-series data*, PLoS One, 8 (2013), p. e66031.

[4] C. Campbell, S. Yang, R. Albert, and K. Shea, *A network model for plant–pollinator community assembly*, Proceedings of the National Academy of Sciences, 108 (2011), pp. 197–202.

[5] K. J. Cios and Ł. A. Kurgan, *Hybrid inductive machine learning: An overview of clip algorithms*, in New Learning Paradigms in Soft Computing, Springer, 2002, pp. 276–322.

[6] T. M. Cover and J. A. Thomas, *Elements of information theory*, John Wiley & Sons, Hoboken, New Jersey, 2 ed., 2006.

[7] J. Czerniak and H. Zarzycki, *Application of rough sets in the presumptive diagnosis of urinary system diseases*, in Artificial intelligence and security in computing systems, Springer, 2003, pp. 41–51.

[8] M. I. Davidich and S. Bornholdt, *Boolean network model predicts cell cycle sequence of fission yeast*, PloS one, 3 (2008), p. e1672.

[9] C. Ding and H. Peng, *Minimum redundancy feature selection from microarray gene expression data*, Journal of bioinformatics and computational biology, 3 (2005), pp. 185–205.

[10] Å. Flobak, A. Baudot, E. Remy, L. Thommesen, D. Thieffry, M. Kuiper, and A. Lægreid, *Discovery of drug synergies in gastric cancer cells predicted by logical modeling*, PLoS computational biology, 11 (2015), p. e1004426.

[11] S. W. Golomb and A. W. Hales, *Hypercube tic-tac-toe*, More Games of No Chance, 42 (2002), pp. 167–180.

[12] S. R. Hegde, H. Rajasingh, C. Das, S. S. Mande, and S. C. Mande, *Understanding communication signals during mycobacterial latency through predicted genome-wide protein interactions and boolean modeling*, PLoS One, 7 (2012), p. e33893.

[13] T. Helikar, N. Kochi, J. Konvalina, and J. A. Rogers, *Boolean modeling of biochemical networks*, The Open Bioinformatics Journal, 5 (2011), pp. 16–25.

[14] I. Irurzun-Arana, J. M. Pastor, I. F. Trocóniz, and J. D. Gómez-Mantilla, *Advanced boolean modeling of biological networks applied to systems pharmacology*, Bioinformatics, 33 (2017), pp. 1040–1048.

[15] L. A. Kurgan, K. J. Cios, R. Tadeusiewicz, M. Ogiela, and L. S. Goodenday, *Knowledge discovery approach to automated cardiac spect diagnosis*, Artificial intelligence in medicine, 23 (2001), pp. 149–169.
[16] H. Lähdesmäki, I. Shmulevich, and O. Yli-Harja, *On learning gene regulatory networks under the boolean network model*, Machine learning, 52 (2003), pp. 147–167.
[17] F. Li, T. Long, Y. Lu, Q. Ouyang, and C. Tang, *The yeast cell-cycle network is robustly designed*, Proceedings of the National Academy of Sciences of the United States of America, 101 (2004), pp. 4781–4786.
[18] S. Liang, S. Fuhrman, and R. Somogyi, *Reveal, a general reverse engineering algorithm for inference of genetic network architectures*, in Pacific Symposium on Biocomputing, vol. 3, 1998, pp. 18–29.
[19] H. Liu, F. Zhang, S. K. Mishra, S. Zhou, and J. Zheng, *Knowledge-guided fuzzy logic modeling to infer cellular signaling networks from proteomic data*, Scientific reports, 6 (2016), p. 35652.
[20] D. MacLean and D. J. Studholme, *A boolean model of the pseudomonas syringae hrp regulon predicts a tightly regulated system*, PloS one, 5 (2010), p. e9101.
[21] S. Marshall, L. Yu, Y. Xiao, and E. R. Dougherty, *Inference of a probabilistic boolean network from a single observed temporal sequence*, EURASIP Journal on Bioinformatics and Systems Biology, 2007 (2007), pp. 5–5.
[22] O. Patashnik, *Qubic: 4× 4× 4 tic-tac-toe*, Mathematics Magazine, 53 (1980), pp. 202–216.
[23] K. Raman, A. G. Bhat, and N. Chandra, *A systems perspective of host–pathogen interactions: predicting disease outcome in tuberculosis*, Molecular BioSystems, 6 (2010), pp. 516–530.
[24] C. E. Shannon, *A mathematical theory of communication*, Bell System Tech. J., 27 (1948), pp. 379–423, 623–656.
[25] S. N. Steinway, M. B. Biggs, T. P. Loughran Jr, J. A. Papin, and R. Albert, *Inference of network dynamics and metabolic interactions in the gut microbiome*, PLoS computational biology, 11 (2015), p. e1004338.
[26] J. Sun and E. M. Bollt, *Causation entropy identifies indirect influences, dominance of neighbors and anticipatory couplings*, Physica D: Nonlinear Phenomena, 267 (2014), pp. 49–57.
[27] J. Sun, C. Cafaro, and E. M. Bollt, *Identifying the coupling structure in complex systems through the optimal causation entropy principle*, Entropy, 16 (2014), pp. 3416–3433.
[28] J. Sun, D. Taylor, and E. M. Bollt, *Causal network inference by optimal causation entropy*, SIAM Journal on Applied Dynamical Systems, 14 (2015), pp. 73–106.
[29] J. Thakar and R. Albert, *Boolean models of within-host immune interactions*, Current opinion in microbiology, 13 (2010), pp. 377–381.
[30] A. Veliz-Cuba, *An algebraic approach to reverse engineering finite dynamical systems arising from biology*, SIAM Journal on Applied Dynamical Systems, 11 (2012), pp. 31–48.
[31] F. Vitali, L. D. Cohen, A. Demartini, A. Amato, V. Eterno, A. Zambelli, and R. Bellazzi, *A network-based data integration approach to support drug repurposing and multi-target therapies in triple negative breast cancer*, PloS one, 11 (2016), p. e0162407.
[32] S. Von der Heyde, C. Bender, F. Henjes, J. Sonntag, U. Korf, and T. Beissbarth, *Boolean erbb network reconstructions and perturbation simulations reveal individual drug response in different breast cancer cell lines*, BMC systems biology, 8 (2014), p. 75.