Cyber Attacks Against the PC Learning Algorithm

Emad Alsuwat, Hatim Alsuwat, Marco Valtorta(✉), and Csilla Farkas

University of South Carolina, Columbia, SC 29208, USA
{Alsuwat,Alsuwath}@email.sc.edu, {Mgv,Farkas}@cec.sc.edu

Abstract. Data integrity is a key requirement for correct machine learning applications, such as Bayesian network structure learning algorithms. This research studies how an adversary could corrupt the PC structure learning algorithm by inserting fake data. We propose a novel measure of strength of links for Bayesian networks. We show how this measure can be used to attack the PC algorithm. We identify two subclasses of data poisoning attacks: (1) model invalidation attacks that arbitrarily break the structure of the Bayesian network model (2) targeted change attacks that achieve a specific structure. We show that model invalidation attacks require only a few "poisoned" data insertions. Targeted attacks are more difficult and require knowledge of the link strengths and a larger number of corrupt data items than the invalidation attack.

Keywords: Adversarial machine learning · Bayesian networks · Data poisoning attacks · The PC algorithm

1 Introduction and Motivation

Machine learning algorithms, including Bayesian Network algorithms, are not secure against adversarial attacks. A machine learning algorithm is a *secure learning algorithm* if it functions well in adversarial environments [5]. Recently, several researchers addressed the problem of attacking machine learning algorithms [5,8,29,34]. *Data poisoning attacks*, which aim to corrupt the machine learning classifier by contaminating the data in the training phase, are considered one of the most important emerging security threats against machine learning systems [24].

Data poisoning attacks against Support Vector Machines (SVMs) [8,10,16, 23,26,35,36] and Neural Networks (NNs) [37] has been studied extensively. However, we found no research on evaluating the vulnerabilities of Bayesian network learning algorithms against adversarial attacks.

In this work, we investigate data poisoning attacks against Bayesian network algorithms. We study two potential attacks against the Bayesian network structure learning algorithms: model invalidation attacks and targeted change attacks. For model invalidation attacks, an adversary poisons the training dataset such

C. Alzate et al. (Eds.): ECML PKDD 2018 Workshops, LNAI 11329, pp. 159–176, 2019.
https://doi.org/10.1007/978-3-030-13453-2_13

that the Bayesian model will be invalid. For targeted change attacks, an adversary poisons the training dataset to achieve a particular goal, such as masking or adding a link in a Bayesian network model.

The main contributions of this paper are the following:

1. We propose two subclasses of data poisoning attacks against the PC structure learning algorithm and establish the difficulty of carrying out the attacks.
2. We define a novel measure of strength of links between variables in Bayesian networks. This measure can be used to find vulnerable structure of the Bayesian model.
3. We evaluate what are the easiest links to break based on the defined link strength measure in Bayesian networks. We also evaluate the most believable ways to add links to achieve a specific goal.
4. We present and justify a plausible process for targeted attacks on Bayesian networks.
5. We have implemented our approach and demonstrated these attacks.

Our experiments show that the PC algorithms is vulnerable to data poisoning attacks. Moreover, even a small number of adversarial data may be sufficient to corrupt the model. Our ongoing work addresses the development of preventive technologies.

The rest of the paper is structured as follows. In Sect. 2, we present an overview of background information. In Sect. 3, we identify model invalidation attacks against the PC algorithm. In Sect. 4, we identify targeted change attacks against the PC learning algorithm. In Sect. 5, we present our link strength measure. In Sect. 6 we present our empirical results. In Sect. 7, we provide conclusions and directions for future work.

2 Background Information

2.1 Bayesian Networks

Bayesian Networks (BNs) are probabilistic graphical models in which vertices represent a set of random variables and arcs represent probabilistic dependencies between vertices. Formally (according to [25]), we say $BN = (G, P)$ is a Bayesian network, where $G = (V, E)$ is a direct acyclic graph (with $V = \{x_1, x_2, ..., x_n\}$ being the set of random variables or nodes, and E being the set of edges or arcs) and P is a joint probability distribution of the random variables, if it satisfies the following Markov condition: every node is conditionally independent of its non-descendants given its parents.

The following factorization of the joint probability distribution of $V = \{x_1, x_2, ..., x_n\}$ into a product of local probability distributions is equivalent to the following Markov property: $P(V) = \prod_{i=1}^{n} P(x_i \mid parent(x_i))$.

The Notion of D-Separation
In a Bayesian network, there are three basic connections among variables as follows [27]: (1) *Serial connections* (also called *pipelined influences*): in a serial connection (shown in Fig. 1a, ignore the dashed link), changes in the certainty of A will affect the certainty B, which in turn will affect the uncertainty of C. Therefore information may flow from node A through B to C, unless there is evidence about B (B is known or *instantiated*). (2) *Diverging connections*: in a diverging connection (shown in Fig. 1b, ignore the dashed links), changes in the certainty of A will affect the certainty B, which in turn will affect the uncertainty of C. Therefore information may flow from node A through B to C, unless there is evidence about B. (3) *Converging connections* (a.k.a. *v-structure*): in a converging connection (shown in Fig. 1c, ignore the dashed links), changes in the certainty of A cannot affect the certainty C through B, and vice versa. Therefore information cannot flow between A and C through B, unless there is evidence about B. The three types of connections in a casual network formulate the definition of *d-separation* (see [27] for the definition of d-separation).

Structure Learning in Bayesian Networks
There are three main approaches to learning the structure of BNs: *constraint-based*, *score-based*, or *hybrid* algorithms. In this work, we focus on constraint-based algorithms, which count on conditional independence tests to determine the DAG of the learned Bayesian network. *The PC algorithm* [32,33] is a constraint-based algorithm for learning the structure of a Bayesian network from data. The PC algorithm follows the theoretical framework of the IC algorithm to determine the structure of causal models [31]. According to [33], the process performed by the PC algorithm to learn the structure of Bayesian networks can be summarized as follows: (i) For every pair of variables, perform statistical tests for conditional independence. (ii) Determine the skeleton (undirected graph) of the learned structure by adding a link between every pair of statistically dependent variables. (iii) Identify colliders (v-structures) of the learned structure (A $\rightarrow$ B $\leftarrow$ C). (iv) Identify derived directions. (v) Randomly, complete orienting the remaining undirected edges without creating a new collider or a cycle. For the implementation of this paper, we used *the Hugin PC algorithm* (by $Hugin^{TM}$ *Decision Engine* [20,28]), "which is a variant of the original PC algorithm due to [33]" [14].

Prior to Posterior Updating
The statement of Bayes' theorem is: For two events A and B, $P(A \mid B) = \frac{P(B|A)P(A)}{P(B)}$, where (i) $P(A \mid B)$ is the conditional probability of event A given event B (called the posterior probability), (ii) $P(B \mid A)$ is the conditional probability of event B given event A (called the likelihood), (iii) $P(A)$ is the marginal probability of event A (called the prior probability), and (iv) $P(B)$ is the marginal probability of event B ($P(B) > 0$) [25].

Bayesian statistics treats parameters as random variables whereas data is treated as fixed. For example, let θ be a parameter, and D be a dataset, then Bayes' theorem can be expressed mathematically as follows:

$P(\theta \mid D) = \frac{P(D|\theta)P(\theta)}{P(D)}$. Since $P(D)$ is constant [19], we can write Bayes' theorem in one of the most useful form in Bayesian update and inference as follows:

$$\begin{aligned} P(\theta \mid D) &\propto P(D \mid \theta) \times P(\theta) \\ Posterior &\propto Likelihood \times Prior \end{aligned} \tag{1}$$

It is convenient mathematically for the prior and the likelihood to be conjugate. A prior distribution is a *conjugate prior* for the likelihood function if the posterior distribution belongs to the same distribution as the prior [30]. For example, the beta distribution is a conjugate prior for the binomial distribution (as a likelihood function).

$$\begin{aligned} P(\theta \mid D) &\propto Binomial(n, \theta) \times Beta(\alpha, \beta) \\ P(\theta \mid D) &\propto Beta(y + \alpha, n - y + \beta) \end{aligned} \tag{2}$$

Equation 2 is the formula that we are going to use in this paper for prior to posterior update. Starting with a prior distribution $Beta(\alpha, \beta)$, we add the count of successes,y, and the count of failures, $n - y$, from the dataset D (where n is total number of entries in D) to α and β, respectively. Thus, $Beta(y+\alpha, n-y+\beta)$ is the posterior distribution.

Link Strengths in Bayesian Networks

Boerlage introduced the concepts of both connection strength and link strength in a binary Bayesian network model [9]. *Connection strength* for any two variables A and B in a Bayesian network model B_1 is defined as measuring the strength between these two variables by testing all possible paths between them in B_1, whereas *link strength* is defined as measuring the strength these two random variables taking into account only the direct edge $A - B$ [9]. Methods for link strengths measurements are not studied sufficiently [11]. We believe that link strength is critical to understand structural vulnerabilities of Bayesian network models. In this paper, we define a novel and computationally not expensive link strength measure.

2.2 Adversarial Machine Learning

Attacks against machine learning systems have been organized by [5,6,13] according to three features: Influence, Security Violation, and Specificity. Influence of the attacks on machine learning models can be either causative or exploratory. Causative attacks aim to corrupt the training data whereas exploratory attacks aim to corrupt the classifier at test time. Security violation of machine learning models can be a violation of integrity, availability, or privacy. Specificity of the attacks can be either targeted or indiscriminate. Targeted attacks aim to corrupt machine learning models to misclassify a particular class of false positives whereas indiscriminate attacks have the goal of misclassifying all false positives.

Evasion attacks [7,12,15,17,34] and Data poisoning attacks [1,8,10,16,22, 23,26,35–37] are two of the most common attacks on machine learning systems [13]. Evasion attacks are exploratory attacks at the testing phase. In an evasion

attack, an adversary attempts to pollute the data for testing the machine learning classifier; thus causing the classifier to misclassify adversarial examples as legitimate ones. Data poisoning attacks are causative attacks, in which adversaries attempt to corrupt the machine learning classifier itself by contaminating the data on training phase.

In this paper, we study the resilience of Bayesian network algorithms, namely the PC algorithm, against data poisoning attacks. To the authors' best knowledge, no study has been performed on evaluating the vulnerabilities of PC algorithm against poisoning attacks. We present the two subclasses of data poisoning attacks against the PC algorithm: (1) Model invalidation attacks and (2) Targeted change attacks.

3 Model Invalidation Attacks

A model invalidation attack against the PC algorithm is a malicious active attack in which adversarial opponents try to corrupt the original model in any way. We demonstrate adversarial attacks to decrease the validation status of the model using the least number of changes. In such an event, adversaries create some formal disturbance in the model. For example, they will try to add imprecise or incorrect data to change the model validation status so that the model is rendered invalid. We distinguish between two ways to invalidate Bayesian network models: (1) Attacks based on the notion of d-separation and (2) Attacks based on marginal independence tests.

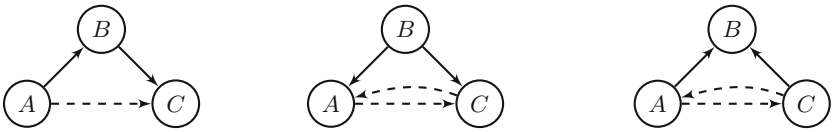
Due to space limitation, we only present selected algorithms in this work. A complete set of algorithms and further details can be accessed in [3,4]. Here is an item list with all the algorithms and short description:

Algorithm	Description
Algorithm 1	Creating a New Converging Connection
Algorithm 2	Breaking an Existing Converging Connection
Algorithm 3	Edge Deleting
Algorithm 4	Removing a Weak Edge
Algorithm 5	Edge adding
Algorithm 6	Adding the Most Believable yet Incorrect Edge
Algorithm 7	Targeted Change Attacks

3.1 Model Invalidation Attacks Based on the Notion of D-Separation

Based on the definition of d-separation, adversaries may attempt to introduce a new link in any triple $(A - B - C)$ in the BN model. This newly inserted link $(A - C)$ will introduce a v-structure in the Bayesian model, thus change the independence relations.

Theorem 1. *Let B_1 and B_2 be two Markov equivalent BNs, and let $<A, B, C>$ be a path in B_1. If a new link is added to B_1 creating B_1', then B_1' and B_2 are not Markov equivalent.*



(a) Adding the dashed link to the serial connection.

(b) Adding one of the dashed links to the diverging connection.

(c) Adding one of the dashed links and shielding collider B.

Fig. 1. Three cases for the proof of Theorem 1.

Proof Sketch. Adding a new edge to the path $<A, B, C>$ in Bayesian network model B_1 affects the Markov equivalence class of B_1 (two Bayesian networks are *Markov equivalent* if and only if they have the same skeleton and the same v-structures (unshielded colliders) [2]). Any sound learning algorithm will try to avoid the occurrence of a cycle; thus, in the triple $(A - B - C)$, either an existing collider is shielded, and a new link is introduced (as shown in Fig. 1c) or a new link is added (as shown in Figs. 1a and b). In either case, the Markov equivalence class of B_1 will be violated.

Within model invalidation attacks based on the notion of d-separation, we can further identify two subclasses:

Creating a New Converging Connection (V-Structure)
Adversarial attackers can corrupt Bayesian network models by introducing a new converging connection. Adversaries will attempt to poison the learning dataset with the goal of introducing a new v-structure by adding a new link to any serial or diverging connection in Bayesian network models. Adding such an edge will not only introduce a new collider but also change the equivalence class of the learned Bayesian network model.

Theorem 2. *Let B_1 be a Bayesian network model, and let $<A, B, C>$ be a path in B_1 with either a serial connection or diverging connection, then introducing a new edge on the path $<A, B, C>$* must *create a new converging connection in B_1.*

Proof Sketch. Trivially follows. [See Figs. 1a and b].

We have developed an algorithm (called *Algorithm 1: Creating a New Converging Connection Procedure*) to tests the resilience of the PC learning algorithm against this type of attacks. Our empirical results are given in Sect. 6.

Breaking an Existing Converging Connection (V-Structure)
Adversaries can exploit Bayesian network models by breaking an existing converging connection. The PC algorithm starts by identifying *unshielded colliders* (v-structure with unmarried parents) when learning the Bayesian network

structure from data [33]; therefore, attacking v-structures will make a significant corruption to the learned BN structures since the learned model will have a different equivalence class than the expected one. Such an adversarial attack can be done by marrying the parents of an unshielded collider. Note that, if vertex B is an *unshielded collider* on the path $<A, B, C>$, then A and C are independent unconditionally, but are dependent conditionally on B in most cases (faithfulness assumption [33]).

Theorem 3. *Let B_1 be a Bayesian network model, and let B be an unshielded collider on the path $<A, B, C>$, then introducing a new edge on the path $<A, B, C>$* must *break the existing converging unshielded connection at vertex B.*

Proof Sketch. Trivially follows. [See Fig. 1c].

We have developed an algorithm (called *Algorithm 2: Breaking an Existing Converging Connection Procedure*) to check the robustness of the PC algorithm against the feasibility of shielding an existing converging connection. Our empirical results are presented in Sect. 6.

3.2 Model Invalidation Attacks Based on Marginal Independence Tests

When learning the structure of a Bayesian network model from data, the PC algorithm starts by analyzing the conditional independence statements between variables. It performs χ^2 statistical test on the given dataset to establish the set of statistical independence statements for the learned causal model [27]. Using this information of how the PC algorithm works, adversarial attackers may contaminate the input dataset with the goal of removing weak edges or adding the most believable yet incorrect links. Based on the direct impact of marginal independence tests on the PC algorithm, model invalidation attacks can be divided into two main types: (1) removing weak edges and (2) adding the most believable yet incorrect edge.

Removing a Weak Edge

We show that it is feasible to use link strengths measure to identify and rank the edges on a causal model from the weakest to the strongest. Thus, adversarial opponents may attempt to poison the learning dataset with the goal of removing weak edges.

We have developed an algorithm (called *Algorithm 4: Removing a Weak Edge Procedure*) to check the resilience of the PC algorithm against attacks that target weak edges. Our algorithm calculates the strength of each link in a Bayesian model and then ranks the edges from the weakest to the strongest edge. It then checks the robustness of the PC algorithm against the feasibility of deleting the weakest edge. Our empirical results are presented in Sect. 6.

Adding the Most Believable yet Incorrect Edge
We show that it is feasible to use link strengths measure to identify and rank the edges on a causal model from the most to the least believable edge. Thus, adversaries can cleverly use data poisoning attacks craft the input dataset to the Bayesian network model so that adding those incorrect yet plausible edges is viable.

We have developed an algorithm (called *Algorithm 6: Adding the Most Believable yet Incorrect Edge Procedure*) to check the robustness of the PC algorithm against this attack. The algorithm starts by learning the structure of the Bayesian network model and then uses the defined link strengths measure to rank a given set of edges that could be added to the learned model from the most to the least believable edge. Our algorithm then checks robustness of the PC algorithm against the feasibility of adding the most believable edge. Our empirical results are presented in Sect. 6.

4 Targeted Change Attacks

A targeted change attack against the PC algorithm is an active malicious attack in which malicious agents try to move from the state of "what I have" to the state of "what I want" by poisoning the learning dataset. Adversaries attempt to plan attacks against Bayesian network models using the least number of changes. That is, they will attempt to move from the existing model to the desired model using the least and inconspicuous number of changes. As such, adversaries assess the difficulty of entering or modifying data that promises to intentionally change the current model into the desired model. By doing so, the adversary is able to make the changed model behave exactly as they want.

A targeted change attack is more harmful and sophisticated than model invalidation attack. For this, adversaries attempt to poison the input dataset aiming for a specific result of the BN model; therefore, it misclassifies a certain class of false positives and false negatives. Before we present *Algorithm 7*, we have developed two algorithms needed for our experiments, *Algorithm 3: Edge Deleting Procedure*, which provides algorithmic details of the robustness of the PC algorithm against the feasibility of deleting an existing edge in a Bayesian network model as follows, and *Algorithm 5: Adding an Edge Procedure*, which checks the robustness of the PC algorithm against the feasibility of introducing a link between two vertices that do not lie in a triple in a BN model.

Algorithm 7. Targeted Change Attacks Procedure

Input : Dataset DB_1 ▷ Original dataset with n cases
Output: Contaminated dataset DB_2 or a failure message

Procedure `Targeted Change Attacks(`DB_1`)`
  Use the PC algorithm for learning the structure of Bayesian network model B_1 from dataset DB_1 (setting the significance of the Hugin PC to the default level, which is 0.05 [20])
  Use L_S to rank the edges of B_1 from the weakest to the strongest edge
  Choose a set of edge Q that could be added to B_1
  Use L_S to rank the set Q from the most to the least believable edge
  Plan a targeted attack (the set of edges to be added or deleted from B_1)
  repeat
    if *there is a need to introduce a new link in* B_1 **then**
      Use *Algorithm 1* to introduce a new v-structure, *Algorithm 2* to break an existing collider, or *Algorithm 5* to add a link between two vertices that do not lie in a triple
    end
    if *there is a need to delete an existing link* **then**
      Use *Algorithm 3*
    end
    if *there is a need to remove the weakest edge* **then**
      Use *Algorithm 4*
    end
    if *there is a need to add the most believable edge* **then**
      Use *Algorithm 6*
    end
  until *the targeted attack is achieved*
end

Algorithm 7 starts by learning the structure of the Bayesian network model B_1 from dataset DB_1. It then uses the defined link strengths measure to rank the edges of B_1 from the weakest to the strongest edge. A malicious user can enter the set of edges Q that the user wants to add to the model B_1. The defined link strength measure is used to rank the set of edge Q from the most to the least believable edge.

The malicious user then plans a targeted change attack. The adversary, in this case, chooses the set of edges that could be added to or deleted from the causal model B_1. For example, an attacker may think it is feasible to achieve his goal by adding a new plausible link and deleting an existing one.

If the attacker wants to add a new link $A - C$ and this new link introduces a new v-structure in a triple $A - B - C$, then *Algorithm 1* is called. On the hand, if the link $A - C$ shield a collider B in a triple $A - B - C$, then *Algorithm 2* is called. Otherwise, *Algorithm 5* is called to add a link between two vertices that do not lie in a triple in a Bayesian network model (see [3] for more algorithmic details about other algorithms).

If the attacker wants to delete an existing edge. There are two algorithms that can check the feasibility of achieving this goal. *Algorithm 3* checks the feasibility of deleting any edge in a Bayesian network model, and *Algorithm 4* checks the feasibility of deleting the weakest edge in a Bayesian network model.

In all different scenarios, *Algorithm 7* returns a contaminated dataset DB_2 if achieving the targeted attack is feasible; otherwise, a failure message will be printed if the number of added cases will be more than $\beta \times n$, where β is *data poisoning rate* at which we are allowed to add new "poisoned" cases to DB_1 (we default set $\beta \leq 0.05$)

5 Measuring Link Strengths from Data in Discrete Bayesian Networks

In this section, we introduce a novel link strength measure between two random variables in a discrete Bayesian network model. It is essential to not only study the existence of a link in a causal model but also define a reliable link strengths measure that is useful in Bayesian reasoning [9,11]. The new defined link strengths measure assigns a number to every link in a Bayesian network model. This number represents the lowest confidence of all possible combinations of assignments of posterior distributions. The defined link strengths measure will guide our edge removal and insertion process. Our novel approach is as follows:

Given a discrete dataset DB_1 and a Bayesian network structure B_1 learned by the PC algorithm using DB_1, for every link $variable_1 \rightarrow variable_2$ in B_1, build a contingency table for the two discrete variables $variable_1$ and $variable_2$ with i and j states, respectively (as shown in Table 1). Table 1 is structured as follows: [the cell's observed counts obtained from DB_1], (the cell's expected counts, calculated as follows: $\frac{Observed\ \ Row\ \ Total \times Observed\ \ Column\ \ Total}{Observed\ \ Grand\ \ Total (denoted\ \ as\ \ n)}$), and <the cell's chi-square test statistic, calculated as follows:$\frac{(n-e)^2}{e}$> [21]. To measure the strength of links of a causal model: (1) we compute the posterior distributions for each link $variable_1 \rightarrow variable_2$ as follows: $P(variable_2 \mid variable_1) = Beta(y+\alpha, n-y+\beta)$ where $variable_2 \mid variable_1$ is all possible combinations of assignments to $variable_2$ and $variable_1$, and then (2) we use our link strength measure (denoted as $L\_S(Variable_1 \rightarrow Variable_2)$), which is defined as follows:

$$L\_S(Variable_1 \rightarrow Variable_2) = \min_{y \in Y}(pdf(\tfrac{y+\alpha}{\alpha+n+\beta})) \tag{3}$$

where $Y = \{n_{11}, n_{12}, \cdots, n_{1j}, n_{21}, n_{22}, \cdots, n_{2j}, \cdots, n_{i1}, n_{i2}, \cdots, n_{ij}\}$, pdf is the probability density function, and $\frac{y+\alpha}{\alpha+n+\beta}$ is the mean of the posterior distribution.

Interpretation: For any two random variables in a causal model ($variable_1$ with i states and $variable_2$ with j states), there are $i \times j$ combinations of assignments of posterior distributions. For every posterior distribution, we have a prior distribution that is a conjugate prior for the likelihood function. For instance, a posterior distribution in the form $Beta(y+\alpha, n-y+\beta)$ has a Beta-distributed prior, $Beta(\alpha, \beta)$, which is a conjugate prior for the likelihood function, $Binomial(n, \theta)$. Considering all $i \times j$ posterior distributions for the two random $variable_1$ and $variable_2$, we can measure the uncertainty of that link by measuring how peaked the posterior distributions (Beta distributions in our experiments) are; thus, we can identify the link strength based on the uncertainty level. The more peaked the posterior distribution is, the more certainty

Table 1. Contingency table for two discrete variables $variable_1$ and $variable_2$ with i and j states, respectively.

	$Variable_2$			
$Variable_1$	$State_{\mathbf{1}}$	$\cdots$	$State_{\mathbf{j}}$	Observed row total
$State_{\mathbf{1}}$	$[n_{11}]$, (e_{11}), $<ts_{11}>$	$\cdots$	$[n_{1j}]$, (e_{1j}), $<ts_{1j}>$	$\sum_{t=1}^{j} n_{1t}$
$\vdots$	$\vdots$	$\cdots$	$\vdots$	$\vdots$
$State_{\mathbf{i}}$	$[n_{i1}]$, (e_{i1}), $<ts_{i1}>$	$\cdots$	$[n_{ij}]$, (e_{ij}), $<ts_{ij}>$	$\sum_{t=1}^{j} n_{it}$
Observed column total	$\sum_{t=1}^{i} n_{t1}$	$\cdots$	$\sum_{t=1}^{i} n_{tj}$	n (Observed grand total)

we have about the posterior distribution probability. In other words, the peak of a beta distribution, $Beta(\alpha', \beta')$, is reached at its mean, $\frac{\alpha'}{\alpha'+\beta'}$. Thus, the peak of the posterior distribution is reached at $\frac{y-\alpha}{n-y+\beta}$. In the defined link strength measure, we define the link strength for any link between two random variables in a causal model as the value of the smallest peak. This point is the point at which the model has seen the fewest number of cases; thus, it is the most critical point through which this link can be manipulated.

We use this measure to identify weak edges (i.e., low values of L_S). These edges are the easiest to remove from a given causal model. We also use the L_S value to identify location for new edges to be added. We claim that the highest L_S value, the most believable the new edge is.

6 Empirical Results

In this section, we demonstrate the robustness of the PC learning algorithm against the proposed data poisoning attacks. The feasibility of such attacks is investigated through empirical results on the Chest Clinic Network [18].

We implemented the Chest Clinic Network using $Hugin^{TM}$ *Research 8.1.* Then we simulated dataset of 10,000 cases for our experiments by using $Hugin^{TM}$ *case generator* [20,28]. We call this dataset as DB_1. Using the PC algorithm on dataset DB_1 with 0.05 significance setting [20], the resulting structure is given in Fig. 3. While the two networks belong to different Markov equivalence classes, we will use the network of Fig. 3 as the starting point of our experiments.

We performed the following three experiments: (1) Model invalidation attacks based on the notion of d-separation. (2) Model invalidation attacks based on marginal independence tests. (3) A targeted attack against the Chest Clinic Network dataset.

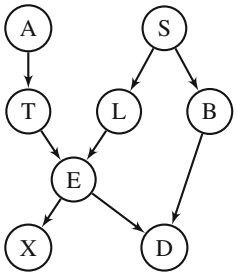


Fig. 2. The original Chest Clinic Network.

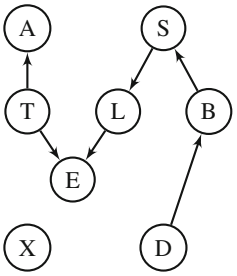


Fig. 3. B_1, the result of feeding DB_1 to the PC algorithm with significance level at 0.05

6.1 Model Invalidation Attacks Based on the Notion of D-Separation

In our first experiment, we evaluated the effectiveness of model invalidation attacks based on the notion of d-separation (Sect. 3.1) to poison the Chest Clinic Network dataset DB_1. Our aim is to introduce a new v-structure. That is, (1) add the links $D - S$, $B - L$ and $S - E$ to the serial connections $D \rightarrow B \rightarrow S$, $B \rightarrow S \rightarrow L$ and $S \rightarrow L \rightarrow E$, respectively, and (2) add the link $A - E$ to the diverging connection $A \leftarrow T \rightarrow E$. We also study the robustness of the PC learning algorithm against the attacks aiming to break an existing v-structure, i.e., to shield the collider $T \rightarrow E \leftarrow L$.

We present our results in Figs. 4, 5, 6, 7, and 8. We succeeded to invalidate (change the Markov equivalence class) the model learned by the PC algorithm. We had to introduce 74 corrupt cases (data items) to introduce the link $D - S$. To introduce links $B - L$, $S - E$, and $A - E$ required 13, 40, and 3 corrupt cases, respectively. To shield the collider E, we only needed 8 poisoning data items. In addition, when we increased the number of corrupted data items, the PC learning algorithm was acting unstably. Our results after adding 17 poising cases to introduce the malicious link $T - L$ is in Fig. 9.

We also observed that the choice of corrupt data items affects the efficiency of the attack. That is, when introducing a malicious link between two random variables, a cell with a higher test statistics value $<ts_{ij}>$ in the contingency table of these two random variables requires fewer corrupt data items than a cell with a lower test statistics value. For example, when poisoning dataset DB_1 to add the link $D - S$, we needed more corrupt data items as the value of test statistics got lower. The results are as follows: the cell with $D = yes$ and $S = yes$ required 74 cases, the cell with $D = yes$ and $S = no$ required 272 cases, the cell with $D = no$ and $S = yes$ required 1120 cases, and the cell with $D = no$ and $S = no$ required 1701 cases. Overall, we showed that the PC algorithm is vulnerable to model invalidation attacks based on the notion of d-separation.

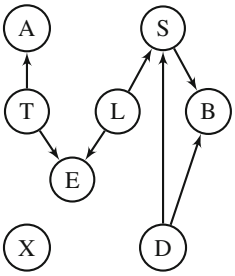


Fig. 4. Introducing a new converging connection in the triple $D - B - S$.

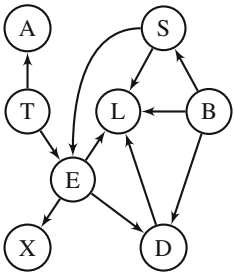


Fig. 5. Introducing a new converging connection in the triple $B - S - L$.

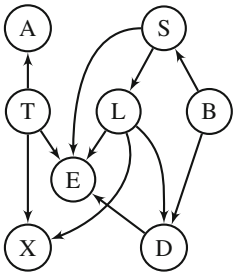


Fig. 6. Introducing a new converging connection in the triple $S - L - E$.

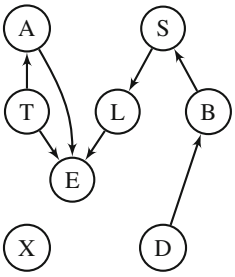


Fig. 7. Introducing a new converging connection in the triple $A - T - E$.

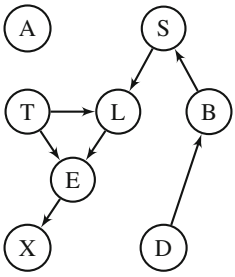


Fig. 8. Breaking an existing converging connection in the triple $T - E - L$.

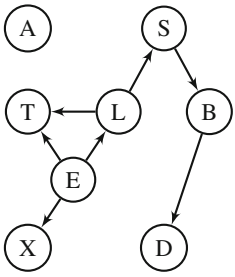


Fig. 9. The result of using 17 cases to break the v-structure $T \rightarrow E \leftarrow L$.

Table 2. Posterior distributions for the Chest Clinic Network.

Link	Posterior distributions (beta distributions)
P(T | A)	Beta(10,99) Beta(106,9789) Beta(99,10) Beta(9789,106)
P(L | S)	Beta(481,4510) Beta(47,4966) Beta(4510,481) Beta(4966,47)
P(B | S)	Beta(3019,1972) Beta(1514,3899) Beta(1972,3019) Beta(3899,1514)
P(E | T)	Beta(115,1) Beta(523,9365) Beta(1,115) Beta(9365,523)
P(E | L)	Beta(527,1) Beta(111,9365) Beta(1,527) Beta(9365,111)
P(D | B)	Beta(3638,895) Beta(725,4746) Beta(895,3638) Beta(4746,725)
P(D | E)	Beta(520,118) Beta(3843,5523) Beta(118,520) Beta(5523,3843)
P(X | E)	Beta(624,14) Beta(454,8912) Beta(14,624) Beta(8912,454)

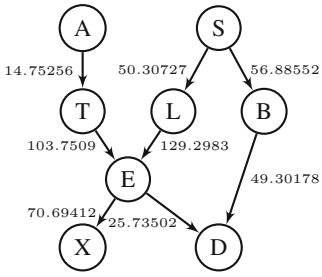


Fig. 10. Results of L_S on the Chest Clinic Network.

6.2 Model Invalidation Attacks Based on Marginal Independence Tests

Link strength measure is needed for the second experiment. For the Chest Clinic Network. Given the Chest Clinic network model as shown in Fig. 2 and the dataset DB_1, we followed the *two steps* presented in Sect. 5. Table 2 contains the posterior distributions calculated in *step 1*. Figure 10 shows the final link strength evaluation (L_S) (calculated in *step 2*).

We will use these strength measures in this section and in Sect. 6.3 to illustrate the ease of removing existing links and adding links to a causal model.

In the second experiment, we evaluated the effectiveness of model invalidation attacks based on marginal independence tests (Sect. 3.2) to poison the Chest Clinic Network dataset DB_1. In this experiment, we check the resilience of the PC algorithm against the feasibility of deleting the weakest edge in the Bayesian model B_1. To determine the weakest edge in B_1, we do the following: (1) use the defined link strength measure L_S to rank the edges of B_1 from the weakest to the strongest edge, and (2) check the feasibility of poisoning dataset DB_1 to remove the weakest edge. We also study the robustness of the PC algorithm against attacks aiming to add the most believable yet incorrect edge to B_1. To determine the most believable edge to be added to B_1, we do the following: (1) determine the set of edges Q that could be added to the model B_1 (in this experiment, we let $Q = \{A - S, T - S, D - S, L - B, L - T\}$), (2) use the defined link strength measure to rank the set of edges Q from the most to the least believable edge, and (3) check the feasibility of poisoning dataset DB_1 to add the most believable edge.

We present our results of deleting the weakest edge from B_1 in Table 3 and Fig. 11. We succeeded to invalidate the model learned by the PC algorithm. We had to modify only 3 cases to break the weakest link $A - T$. Our results of adding the most believable edge to B_1 are presented in Tables 4, 5, and Fig. 12. We succeeded to fool the PC algorithm and invalidate the learned model. We had to introduce only 13 corrupt data items to add the most believable link $B - L$.

We observed that when removing an edge from a causal model, the choice of corrupt data items has an impact on the efficiency of the attack. That is, transferring data items from the cell with the highest test statistics value to the cell with the lowest test statistics value in a contingency table of two random variables will accelerate the process of removing the link between them. Overall, we showed that the PC algorithm is vulnerable to model invalidation attacks based on marginal independence tests.

Table 3. The result of using L_S to rank B_1 edges from the weakest to the strongest.

Link	Link strength L_S	Rank
$A \rightarrow T$	14.75256	1
$S \rightarrow L$	50.30727	3
$S \rightarrow B$	56.88552	4
$T \rightarrow E$	103.7509	5
$L \rightarrow E$	129.2983	6
$B \rightarrow D$	49.30178	2

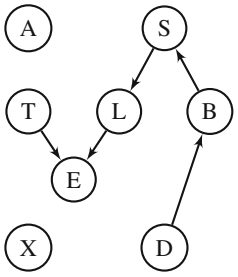


Fig. 11. The result of removing the weakest link in B_1, $A \rightarrow T$

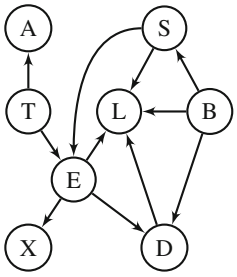


Fig. 12. The result of adding the most believable link to B_1, $B \rightarrow L$.

Table 4. Posterior distributions for the set of edges Q.

Link	Posterior distributions (beta distributions)
P(S | A)	Beta(57, 52) Beta(4934, 4961) Beta(57, 52) Beta(4934, 4961)
P(T | S)	Beta(49, 4942) Beta(67, 4946) Beta(49, 4942) Beta(67, 4946)
P(D | S)	Beta(2728, 2263) Beta(1635, 3378) Beta(2728, 2263) Beta(1635, 3378)
P(L | B)	Beta(312, 4221) Beta(216, 5255) Beta(312, 4221) Beta(216, 5255)
P(L | T)	Beta(5, 111) Beta(523, 9365) Beta(5, 111) Beta(523, 9365)

Table 5. L_S results.

Link	{Link strength L_S}	Rank
$A \rightarrow S$	8.313748	5
$S \rightarrow T$	28.66903	3
$S \rightarrow D$	54.90557	2
$B \rightarrow L$	91.51039	1
$T \rightarrow L$	21.92398	4

6.3 A Targeted Attack Against the Chest Clinic Network Dataset

A further goal of this research is to study the influence of targeted change attacks on our dataset DB_1. We validate the effectiveness of targeted change attacks described in *Algorithm 7* (Sect. 4) to poison the Chest Clinic network dataset DB_1 with the goal of achieving a particular change to the model. *Algorithm 7* checks the robustness of the PC algorithm against the feasibility of implementing a targeted change attack.

Given the link strength measure L_S for ranking the edges of the model B_1 from the weakest to the strongest edge (Table 3) and given L_S for ranking the set of edges Q that could be added to the model B_1 from the most to the least believable edge (Table 5), we aim to change **model B_1 such that it concludes that smoking (S) causes dyspnoea (D) but not lung cancer(L)**. Our attack had the following two steps: step (1) use *Algorithm 7* to delete the link $S \rightarrow L$, and then step (2) use *Algorithm 7* again to add the link $S \rightarrow D$ (Fig. 13).

We present our results in Figs. 14, and 15. We observed that *Algorithm 7* succeeded to delete the link $S \rightarrow L$ by modifying only 114 data items in our dataset DB_1, resulting in a dataset DB_2 (Fig. 14). Then we fed DB_2 to *Algorithm 7* succeeded to add the link $D \rightarrow S$. We needed only 74 cases to introduce the link $D \rightarrow S$ in dataset DB_2 (Fig. 15). Overall, we showed that the PC algorithm is vulnerable to targeted change attacks.

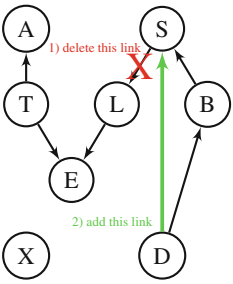


Fig. 13. A targeted attack against the model B_1

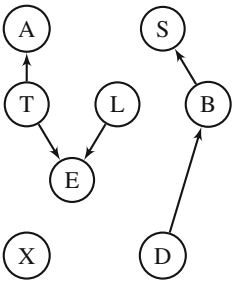


Fig. 14. The model B_1 after achieving *step 1* (deleting $S \rightarrow L$)

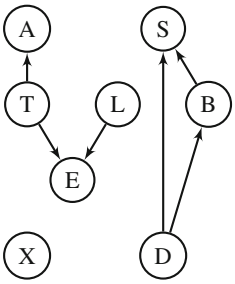


Fig. 15. The model B_1 after achieving the *two steps* of the targeted attack

7 Conclusion and Future Work

As machine learning techniques become more pervasive, it is important to be aware of the danger of malicious attackers based on introducing corrupted data items. We explored the vulnerabilities of a commonly used structural learning algorithm for BNs to adversarial attacks. To carry out experiments, we define a novel measure of link strength. Our results indicate that a malicious attacker can both invalidate the model and modify it according to a desired aim with relatively few data items. The experiments presented in this paper involve a commonly used synthetic Bayesian network. Our ongoing work develops prevention and detection methods against such adversarial attacks. We also aim to acquire a real world dataset for future experiments.

References

1. Alfeld, S., Zhu, X., Barford, P.: Data poisoning attacks against autoregressive models. In: AAAI, pp. 1452–1458 (2016)
2. Ali, A.R., Richardson, T.S., Spirtes, P.L., Zhang, J.: Towards characterizing Markov equivalence classes for directed acyclic graphs with latent variables. arXiv preprint arXiv:1207.1365 (2012)
3. Alsuwat, E., Valtorta, M., Farkas, C.: Bayesian structure learning attacks. University of South Carolina, SC, USA, Technical report (2018)
4. Alsuwat, E., Valtorta, M., Farkas, C.: How to generate the network you want with the pc learning algorithm. Proc. WUPES **18**, 1–12 (2018)
5. Barreno, M., Nelson, B., Joseph, A.D., Tygar, J.D.: The security of machine learning. Mach. Learn. **81**(2), 121–148 (2010). https://doi.org/10.1007/s10994-010-5188-5
6. Barreno, M., Nelson, B., Sears, R., Joseph, A.D., Tygar, J.D.: Can machine learning be secure? In: Proceedings of the 2006 ACM Symposium on Information, Computer and Communications Security, pp. 16–25. ACM (2006)

7. Biggio, B., et al.: Evasion attacks against machine learning at test time. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) ECML PKDD 2013. LNCS (LNAI), vol. 8190, pp. 387–402. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40994-3_25
8. Biggio, B., Nelson, B., Laskov, P.: Poisoning attacks against support vector machines. In: Proceedings of the 29th International Conference on International Conference on Machine Learning, pp. 1467–1474. Omnipress (2012)
9. Boerlage, B.: Link strength in Bayesian networks. Ph.D. thesis, University of British Columbia (1992)
10. Burkard, C., Lagesse, B.: Analysis of causative attacks against SVMs learning from data streams. In: Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics, pp. 31–36. ACM (2017)
11. Ebert-Uphoff, I.: Tutorial on how to measure link strengths in discrete Bayesian networks. Technical report, Georgia Institute of Technology (2009)
12. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
13. Huang, L., Joseph, A.D., Nelson, B., Rubinstein, B.I., Tygar, J.: Adversarial machine learning. In: Proceedings of the 4th ACM Workshop on Security and Artificial Intelligence, pp. 43–58. ACM (2011)
14. Hugin Expert A/S: HUGIN Researcher API 7.0 (2008). https://www.hugin.com/
15. Kantchelian, A., Tygar, J., Joseph, A.: Evasion and hardening of tree ensemble classifiers. In: International Conference on Machine Learning, pp. 2387–2396 (2016)
16. Koh, P.W., Liang, P.: Understanding black-box predictions via influence functions. In: International Conference on Machine Learning, pp. 1885–1894 (2017)
17. Laskov, P., et al.: Practical evasion of a learning-based classifier: a case study. In: 2014 IEEE Symposium on Security and Privacy (SP), pp. 197–211. IEEE (2014)
18. Lauritzen, S.L., Spiegelhalter, D.J.: Local computations with probabilities on graphical structures and their application to expert systems. J. R. Stat. Soc. Ser. B (Methodol.) **50**, 157–224 (1988)
19. Lynch, S.M.: Introduction to Applied Bayesian Statistics and Estimation for Social Scientists. Springer, Heidelberg (2007). https://doi.org/10.1007/978-0-387-71265-9
20. Madsen, A.L., Jensen, F., Kjaerulff, U.B., Lang, M.: The hugin tool for probabilistic graphical models. Int. J. Artif. Intell. Tools **14**(03), 507–543 (2005)
21. McHugh, M.L.: The chi-square test of independence. Biochem. Med.: Biochem. Med. **23**(2), 143–149 (2013)
22. Mei, S., Zhu, X.: The security of latent Dirichlet allocation. In: Artificial Intelligence and Statistics, pp. 681–689 (2015)
23. Mei, S., Zhu, X.: Using machine teaching to identify optimal training-set attacks on machine learners. In: AAAI, pp. 2871–2877 (2015)
24. Muñoz-González, L., et al.: Towards poisoning of deep learning algorithms with back-gradient optimization. In: Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security, pp. 27–38. ACM (2017)
25. Neapolitan, R.E., et al.: Learning Bayesian Networks, vol. 38. Pearson Prentice Hall, Upper Saddle River (2004)
26. Newell, A., Potharaju, R., Xiang, L., Nita-Rotaru, C.: On the practicality of integrity attacks on document-level sentiment analysis. In: Proceedings of the 2014 Workshop on Artificial Intelligent and Security Workshop, pp. 83–93. ACM (2014)
27. Nielsen, T.D., Jensen, F.V.: Bayesian Networks and Decision Graphs. Springer, Heidelberg (2009). https://doi.org/10.1007/978-0-387-68282-2

28. Olesen, K.G., Lauritzen, S.L., Jensen, F.V.: aHUGIN: a system creating adaptive causal probabilistic networks. In: Uncertainty in Artificial Intelligence, pp. 223–229. Elsevier (1992)
29. Papernot, N., McDaniel, P., Goodfellow, I.: Transferability in machine learning: from phenomena to black-box attacks using adversarial samples. arXiv preprint arXiv:1605.07277 (2016)
30. Raiffa, H., Schlaifer, R.: Applied statistical decision theory. Graduate School of Business Administration, Harvard University, Division of Research (1961)
31. Scutari, M.: Learning Bayesian networks with the bnlearn R package. J. Stat. Softw. **35**(3), 1–22 (2010)
32. Spirtes, P., Glymour, C.: An algorithm for fast recovery of sparse causal graphs. Soc. Sci. Comput. Rev. **9**(1), 62–72 (1991)
33. Spirtes, P., Glymour, C.N., Scheines, R.: Causation, Prediction, and Search. MIT Press, Cambridge (2000)
34. Szegedy, C., et al.: Intriguing properties of neural networks. arXiv preprint arXiv:1312.6199 (2013)
35. Xiao, H., Xiao, H., Eckert, C.: Adversarial label flips attack on support vector machines. In: ECAI, pp. 870–875 (2012)
36. Xiao, H., Biggio, B., Nelson, B., Xiao, H., Eckert, C., Roli, F.: Support vector machines under adversarial label contamination. Neurocomputing **160**, 53–62 (2015)
37. Yang, C., Wu, Q., Li, H., Chen, Y.: Generative poisoning attack method against neural networks. arXiv preprint arXiv:1703.01340 (2017)