4th Workshop on Causal Inference and Machine Learning for Intelligent Decision Making

Long Beach CA, December 8th, 2017

Obtaining Accurate Probabilistic Causal Inference by Post-Processing Calibration

Fattaneh Jabbari[1], **Mahdi Pakdaman Naeini**[2], **Gregory Cooper**[1,3]
[1]Intelligent Systems Program, University of Pittsburgh, Pittsburgh, PA
[2]Paulson School of Engineering and Applied Sciences, Harvard University, Cambridge, MA
[3]Department of Biomedical Informatics, University of Pittsburgh, Pittsburgh, PA
fattaneh.j@pitt.edu, pakdaman@seas.harvard.edu, gfc@pitt.edu

Abstract

Discovery of an accurate causal Bayesian network structure from observational data can be useful in many areas of science. Often the discoveries are made under uncertainty, which can be expressed as probabilities. To guide the use of such discoveries, including directing further investigation, it is important that those probabilities be well-calibrated. In this paper, we introduce a novel framework to derive calibrated probabilities of causal relationships from observational data. The framework consists of three components: (1) an approximate method for generating initial probability estimates of the edge types for each pair of variables, (2) the availability of a relatively small number of the causal relationships in the network for which the truth status is known, which we call a *calibration training set*, and (3) a calibration method for using the approximate probability estimates and the calibration training set to generate calibrated probabilities for the many remaining pairs of variables. We also introduce a new calibration method based on a shallow neural network. Our experiments on simulated data support that the proposed approach improves the calibration of causal edge predictions. The results also support that the approach often improves the precision and recall of predictions.

1 Introduction

Much of science consists of discovering and modeling causal relationships in nature. Increasingly, scientists have available multiple complex data and a large number of samples, each of which has an enormous number of measurements recorded, thanks to rapid advancements in sophisticated measurement technology, where this data are often purely observational. In past 25 years, there has been tremendous progress in developing computational methods for discovering causal knowledge from observational data [11, 24, 19, 23, 12]. A primary use of such methods is to analyze observational data to generate novel causal hypotheses that are likely to be correct when subjected to experimental validation; such an approach can significantly increase the efficiency of causal discovery in science.

To make informed decisions about which novel causal hypotheses to investigate experimentally, scientists need to know how likely the hypotheses are to be true. In probabilistic terms, this means they need to have the probabilities of the hypotheses (as output by a causal discovery algorithm) be well-calibrated. Informally, we say that probabilities are well-calibrated if events predicted to occur with probability p do occur about p fraction of the time, for all p [4]. In general, it is important to use calibrated probabilities when making decisions using decision theory.

In this paper, we focus on the discovery of causal Bayesian network (CBN) structure from observational data. In particular, we focus on the discovery of the causal relationships (edge types) between pairs of measured variables. If a causal arc is novel and important, it may be worthwhile to experimentally investigate it. The extent to which it is worth doing so depends in part on how high is

31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA.

the calibrated probability that the causal arc is present. We introduce a method to calibrate edge type probabilities in CBNs with thousands of measured variables and arbitrarily many latent variables.

The method requires the following components: (1) a method for generating initial **probability estimates** of the edge types for each pair of variables; in general those estimates need not be well-calibrated, (2) the truth status of a small unbiased sample of the causal relationships in the network, which we call a **calibration training set**, and (3) a **calibration method** for using the uncalibrated probability estimates and the calibration training set to generate calibrated probabilities for the large number of remaining pairs of variables.

We use a bootstrapping method [8] for generating probability estimates of edge types. This method resamples a dataset n times with replacement and learns a model for each dataset. In particular, for each dataset we use the Really Fast Causal Inference (RFCI) algorithm [3] to estimate the underlying generative network when allowing for the possibility of latent confounders. For any given pair of nodes (A, B), the probability they have a given edge type (e.g., $A \rightarrow B$) is estimated as the fraction of that edge type for (A, B) in the n networks. Previously, researchers have successfully applied this approach for estimating the probabilities of edge types in Bayesian networks [9]. These bootstrap estimates are not guaranteed to represent calibrated posterior probabilities, however, even in the large sample limit of the number of bootstrap samples. A key reason is that heuristic search, while practically necessary, may get stuck in local maxima. Thus, there is a need to map those estimates to calibrated probabilities, which is the focus of the current paper.

The bootstrapping approach described above provides empirical estimates of edge-type posterior probabilities for both constraint-based (e.g., PC, FCI [23], and RFCI [3]) and Bayesian structure learning algorithms (e.g., GES [2]). Bayesian model averaging [15, 10, 14, 7, 13] provides an alternative approach for estimating edge probabilities. However, such Bayesian methods are typically applicable when using datasets in which the number of random variables is in the double digits (for exact search methods) to triple digits (for heuristic search methods). In contrast, we are interested in providing calibrated estimates of edge probabilities for datasets that may contain thousands of variables, as typically encountered with modern biological data. We also note that Bayesian model averaging methods are sensitive to the method applied for heuristic search [9] and to the structure and parameter priors that are used, even if they are non-informative. Consequently, their generated probabilities are still subject to possibly being uncalibrated. Finally, there are no computationally tractable Bayesian methods for discovering CBNs that contain more than a few latent confounders; in contrast, constraint-based methods exist that can perform discovery of CBNs with hundreds (or more) latent variables on datasets with thousands of variables in a feasible amount of time [3].

We assume the availability of a calibration training set that allows us to induce a mapping from bootstrap probability estimates to calibrated posterior probabilities. The training set should contain the truth status for the subset of edge types. In the domain of biomedical applications, the truth status might come, for example, from results published in the literature. We emphasize that the calibration training set can be very small, relative to the number of total node pairs. In the experiments we performed, it consists of less than 0.02% of all the node pairs. Using it, our goal is to generate better calibrated probabilities for the remaining 99.98% of node pairs. In an application using biomedical data, for example, a biomedical scientist who chose to experimentally test causal relationships that have high probabilities (i.e., close to 1) that are well-calibrated could be confident that the experiments would usually corroborate those relationships. We introduce a new neural-network-based calibration method that uses the calibration training set to construct a mapping from bootstrap probability estimates to calibrated posterior probabilities of edge types for all node pairs in a CBN (except those few that are used for training). We apply that mapping to all of those node pairs.

In this paper, we use simulated data to investigate two main questions[1]. First, how calibrated are the bootstrap-derived probabilities of edge types? Second, how calibrated are the probabilities produced by our neural-network-based calibration method? Given a finite calibration training set, the latter method is not guaranteed to always output perfectly calibrated probabilities either. *Our main hypothesis in this paper is that this calibration method will output probabilities that are better calibrated than are the bootstrap probabilities, while being at least as discriminative in terms of measures such as precision, recall, and F1 score.*

[1] Note that it is difficult to obtain gold standards for the causal relationships among the variables in large observational datasets. As a result, the use of simulated data is important and commonly done to evaluate causal discovery methods.

2 Method

In this section we briefly describe the RFCI search, bootstrap RFCI, and calibration model.

2.1 Overview of RFCI

Colombo et al.[3] developed an algorithm called Really Fast Causal Inference (RFCI), which identifies the causal structure of the data-generating process in the presence of latent variables using Partial Ancestral Graphs (PAGs) as a representation. A PAG encodes a Markov equivalence class of Bayesian networks (possibly with latent variables) that exhibit the same conditional independence relationships. RFCI has two stages: (1) adjacency search: this involves a selective search for the (in)dependencies among the measured variables, (2) orientation phase: this orients the endpoints among pairs of nodes that are connected according to the first stage. As is typical of constraint-based causal discovery algorithms, RFCI outputs a single graph structure (PAG) and does not provide any information about the uncertainty of the edges between the nodes in the structure.

2.2 Bootstrap RFCI

Considering the PAG generated by RFCI, it is possible to partition all pairs of nodes (A, B) into the following seven classes: (1) $A \cdots B$: there is no edge between A and B; (2) $A \rightarrow B$: a directed edge from A to B means that A is a direct or indirect cause of B; (3) $B \rightarrow A$: this is similar to (2); (4) $A \circ\!\!\rightarrow B$: this edge type indicates either A is a cause of B, there is an unmeasured confounder of A and B, or both; (5)$B \circ\!\!\rightarrow A$: this is similar to (4); (6) $A \circ\!\!-\!\!\circ B$: this edge type expresses that A is a cause of B, B is a cause of A, there is an unmeasured confounder of A and B, or that there is an unmeasured confounder and one of those two causal relationships holds; (7) $A \leftrightarrow B$: a bi-directed edge between A and B represents the presence of an unmeasured confounder of A and B.

The bootstrap RFCI (BRFCI) method that we apply has three main steps. First, it performs bootstrap sampling over the training data n times ($n = 200$ in our experiments) to create n different bootstrap training datasets. In the second step, it runs RFCI on each of n datasets to obtain n PAGs. Finally, for every pair of nodes, it uses the frequency counts of each edge class for that pair over the generated PAGs to determine a probability distribution for the seven possible edge classes. As mentioned, these bootstrap estimates are not guaranteed to be calibrated. In the following section, we describe a post-processing method to map the bootstrap probabilities to calibrated probabilities.

2.3 Calibration Model

For a pair of nodes (A, B), the resulting output of the BRFCI method will be seven jointly exhaustive and mutually exclusive class probabilities that correspond to the seven classes described above. Therefore, we need to apply a calibration method that post-processes a multi-class classification score (in our case seven classes). One simple approach to devise such a multi-class calibration model is to use a well-performing non-parametric binary classifier calibration method such as isotonic regression [25], averaging over Bayesian binning (ABB) [16], or Bayesian binning into quantiles (BBQ) [17] to post-process the corresponding output probabilities of each class separately. This is performed in a one-versus-remainder fashion as described in [25]. The major drawback of this approach is that such binary calibration methods are histogram-based non-parametric and they require a considerable amount of data to produce well-calibrated probabilities. However, it is often too expensive or not feasible to obtain the truth status for a large number of node pairs in real applications of causal discovery. Consequently, the availability of only a small calibration training set is a critical constraint in the design of the calibration approach.

To resolve this problem, we make a simple extension to Platt's method [20], which is a parametric binary classifier calibration approach. Platt's method uses a sigmoid transformation to map the output of a binary classifier into a calibrated probability. It then uses a logistic loss function to learn the two parameters of the model. The method has two advantages: (1) it has only two parameters that make it a viable choice for low sample size calibration datasets, and (2) the method runs in $O(1)$ at test time, and thus, it is fast. A natural extension to Platt's method for the multi-class calibration task is to use a combination of a softmax transfer function and a cross-entropy loss function instead of a sigmoid function and a logistic loss function, respectively. Minimizing the cross entropy is equivalent to minimizing the empirical Kullback-Leibler divergence of the estimated probabilities and the observed

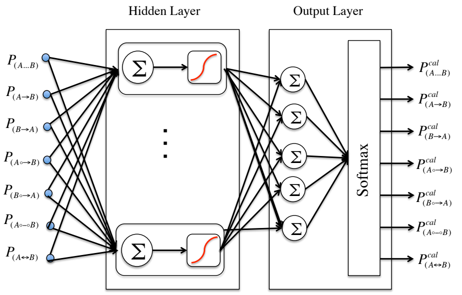
Hidden Layer
Output Layer
$P_{(A...B)}$
$P_{(A\to B)}$
$P_{(B\to A)}$
$P_{(A\circ\!\!\to B)}$
$P_{(B\circ\!\!\to A)}$
$P_{(A\circ\!\!-\!\!\circ B)}$
$P_{(A\leftrightarrow B)}$
Σ
Softmax
$P^{cal}_{(A...B)}$
$P^{cal}_{(A\to B)}$
$P^{cal}_{(B\to A)}$
$P^{cal}_{(A\circ\!\!\to B)}$
$P^{cal}_{(B\circ\!\!\to A)}$
$P^{cal}_{(A\circ\!\!-\!\!\circ B)}$
$P^{cal}_{(A\leftrightarrow B)}$

Figure 1: The structure of post-processing calibration method. The inputs on the left are the bootstrap probabilities for seven edge types. The outputs on the right are the corresponding post-processed probabilities that are intended to be better calibrated.

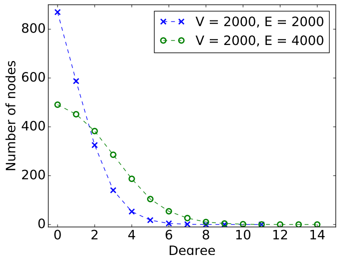

Figure 2: Parent size distribution for simulated CBNs with $V = 2000$ nodes and $E = 2000$ or 4000 edges.

ones. The minimum will be achieved by the true probability distribution and minimizing the cross entropy function will result in finding the closest distribution parameterized by the model to the observed distribution of data [18].

The model that uses the softmax transfer function and optimizes the cross entropy loss function is called softmax regression [18]. The softmax regression-based calibration model inherits the desirable properties of Platt's method. However, similar to Platt's method, the mapping that the softmax regression-based calibration method can learn is restrictive since the final separating boundaries between each pair of classes are always linear. A simple relaxation of this restriction is to use a shallow neural network with one hidden layer. Figure 1 shows the architecture of the shallow neural network model that we use to post-process the bootstrap-generated probabilities.

In our experiments, we train 10 different such shallow neural networks by setting the number of neurons in the hidden layer to be 4, 5, 6, or 7 randomly. At test time, we use the average of the 10 different outputs generated by these models as the final calibrated probability estimates. The averaging is helpful since it reduces the variance error of the predictions and improves the final performance of the post-processed probabilities [5]. We will use the notation $f_{cal}(p1, p2, p3, p4, p5, p6, p7)$ to denote the mapping from a seven-element vector of uncalibrated probabilities that are input to a seven-element vector of calibrated probabilities that are output. We implemented our model using the scikit flow Python package[2], which uses the tensorflow machine learning package [1]. We used the cross-entropy loss function and the adagrad optimization method [6] to learn the parameters; we set the learning rate and the batch size to be 0.1 and 10, respectively.

3 Experimental Methods

This section describes the experimental methods that we used to evaluate the performance of the calibrated network discovery method introduced above. The evaluation involves the following steps:

1. Create a random causal Bayesian network, BN, with V real-valued nodes and E edges. We set V to be 1000 and 2000. We also set the average number of edges per node to be 1 and 2 (i.e., $E = V$ and $E = 2V$). To construct the BN, we first ordered the nodes. Then, we randomly added edges in a forward direction until obtaining the specified mean graph density. This process generates a graph with a power-law-type distribution over the number of parents, with some nodes having many more than the average number of parents (Figure 2). In each BN, the nodes correspond to continuous random variables where for every pair of nodes, (A, B), we parametrize a relation $A \rightarrow B$ in the BN as a structural equation model (SEM): $A = \epsilon_A$ and $B = A\beta + \epsilon_B$, where ϵ_A and ϵ_B are zero-mean Gaussian noise terms and β is a linear coefficient. In our experiments, similar to Ramsey[21, 22], variances of ϵ_A and ϵ_B are uniformly randomly chosen from the interval $[1.0, 3.0]$ and β is drawn uniformly randomly from the interval $[-1.5, 0.2] \cup [0.2, 1.5]$. This choice of parameter values for the simulations implies that, on average, around half of the variance of the variables is due to the error term, which makes structure learning more difficult [21, 22].

2. Simulate a dataset D of size 1000 from BN, subject to constraints that are described below.

[2] https://github.com/tensorflow/skflow

3. Set a percentage of variables, h, to be unobserved (i.e., latent). These latent variables are randomly chosen from confounder variables (i.e., common causes) in a given data-generating BN. We set h to be either 10% or 20%.

4. Generate 200 bootstrap datasets, $DB[1..200]$ from D.

5. For each bootstrap dataset, learn a PAG using the RFCI method; let $PAG[1..200]$ designate these PAGs. RFCI uses Fisher's Z test to check conditional independence of variables in the dataset. We set the significance level at which independence judgments were made to be $\alpha = 0.001$ and 0.005.

6. For each node pair (A, B), calculate the probability distribution $P_e(A, B)$ of the edge types of (A, B) using maximum likelihood estimates on the counts in $PAG[1..200]$.

7. Perform a stratified random sampling on the node pairs to obtain N training samples for calibration and use the rest of the data for testing. We set $N = 70, 140,$ or 210. In obtaining N samples, we used stratified random sampling to select N/7 samples for each of the seven edge classes[3]. In particular, we first sorted the probability scores of edges in each edge class according to the bootstrap probabilities. We then partitioned the instances into 5 bins of $\{[0, 0.2), [0.2, 0.4), [0.4, 0.6), [0.6, 0.8), [0.8, 1]\}$ based on their bootstrap probabilities. Finally, we sampled separately from each bin with equal frequency.

8. Learn the calibration function f_{cal} using the calibration training data.

9. For each node pair (A, B) in the test set, derive $P_e^{cal}(A, B) = f_{cal}\left(P_e(A, B)\right)$.

10. Compare the performance of P_e^{cal} versus P_e in correctly predicting the data-generating structure of BN for the test set pairs, and doing so in a manner that is well calibrated.

In running the above evaluation procedure, step (5) is the most time consuming part that involves running RFCI on 200 bootstrap datasets. However, this is still feasible due to the run-time efficiency of the RFCI method and our use of parallel computing[4]. For all simulations, we used Tetrad[5], which is an open-source, freely available software application that is coded in Java.

Steps (1) through (10) above were repeated for 10 randomly generated BNs and the performance results were averaged. For a given node pair, we take the predicted edge type for that pair to be the one that has the highest probability. Note that although there are seven different edge classes, we consider only five edge types for performance evaluation, because $A \to B$ and $B \to A$ are both directed edge types, and $A \circ\!\!\to B$ and $B \circ\!\!\to A$ are both partially directed edge types.

The first two edge-type-based evaluation measures are *precision* (P) and *recall* (R). To compute these measures for each edge type, we calculated the four basic statistics of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN) for each of the types separately. Precision is then derived as the ratio $TP/(TP + FP)$. Recall is derived as the ratio $TP/(TP + FN)$. We also report the F1-score (i.e., the harmonic mean of the precision and recall), which is a summary measure that shows the overall performance of the predictions in terms of both precision and recall.

We also evaluated the edge-type-based predictions in terms of maximum calibration error (MCE) [17]. We calculated the MCE for each edge type by partitioning the output space of the estimated edge-type probabilities, which is on the interval [0, 1], into equal-frequency bins with 100 randomly chosen instances. The estimated probability for each instance is located in one of the bins. For each bin, we define the associated calibration error as the absolute difference between the mean value of the predictions and the actual observed frequency of positive instances. The MCE calculates the maximum calibration error over all the bins. The lower the value of MCE, the better the calibration of the probability scores. The lowest possible value of MCE is 0 and the highest possible value is 1.

We also report the overall (micro averaged) MCE as a summary measure which shows the performance of the predictions in terms of calibration. To compute this measure, we augmented the seven-element probability distribution vectors, $P_e(A, B)$ for all test instances to form an aggregated vector P_{all}. We also augmented their corresponding 1-of-k binary labels [18, 1] to form an aggregated binary vector Z_{all}. The overall MCE is defined as the maximum calibration error calculated based on P_{all} and Z_{all}.

[3] Using stratified random sampling is crucial due to severe class imbalance of the data (i.e., more than 99% of the pairs are no-edge type).

[4] The running times of the experiments varied from 11 to 176 minutes on a 16-core compute node, which is computationally feasible, because step (5) needs to be done one time only.

[5] https://github.com/cmu-phil/tetrad

4 Experimental Results

This section presents the results of our experiments in evaluating the performance of the generated probabilities for the five edge types before and after calibration. We use the shallow neural network calibration method to learn the calibration function f_{cal} from calibration training data. Since the purpose of this paper is not to compare calibration methods, we do not report the results of experiments on using other calibration methods (e.g., IsoReg or Platt's method). Rather, we only report the results of calibration using the neural network method which we found performs well with relatively small calibration training sample sizes compared to the other calibration methods that we tried.

For each set of configurations (e.g., $N = 210$, $V = 2000$, $E = 2000$), we report the average results of using 10 randomly simulated CBNs. Tables 1a, 1b, 1c, and 1d show the results for CBNs with 2000 nodes (due to the page limit, the results of experiments for $V = 1000$ are not included but similar results are achieved). In these tables, boldface indicates the results that are statistically significantly superior, based on a two-sided Wilcoxon signed rank test at 5% significance level. Tables 1a-1d indicate that by post-processing the bootstrap probabilities, we can improve the overall edge-type performance both in terms of discrimination and calibration. The only exception is the $A \leftrightarrow B$ edge type for which we lose discrimination. This is happening because the original precision and recall of the bootstrap probabilities are very low for this edge type. Consequently, we often obtain very few positive instances from this edge type in the calibration training set, which negatively affects the performance of predictions after calibration. Note that for the no-edge type we do not report precision, recall, and F1, because they were always very close to 1.

Figure 3 shows the calibration diagram [4] of the estimated probabilities before and after calibration when we use 210 calibration training instances. We emphasize that observing 210 calibration instances is equivalent to observing less than 0.02% of all node pairs in the CBN (i.e., there are 1999×10^3 node pairs in a CBN with 2000 nodes). To draw the calibration diagrams, we partitioned the output space of the estimated probabilities into five equal-size bins. In each bin, we draw the average frequency of positive class versus the mean of the predictions that are located in that bin. In these diagrams, the straight dashed line connecting (0, 0) to (1, 1) represents a perfectly calibrated model. The closer a calibration curve is to this line, the better calibrated is the prediction model.

Figure 3 shows the proposed shallow neural network post-processing method often improves the calibration performance of the predictions for the $A \rightarrow B$ edge-type, which is the most important edge type since it is the one that is most likely to drive experimentation. In particular, directed edges that a scientist considers to be high probability, as well as novel and important, would be prime candidates for experimental validation. Furthermore, for this edge-type, the high probability region is arguably the most critical one for making decisions about which directed arcs to investigate, such that false positive experimental investigations are minimized.

Also, the associated diagrams of the no-edge type (i.e., $A \cdots B$) in Figure 3 show that the estimated probabilities are pretty well-calibrated after calibration. This is an interesting observation considering the fact that the precision and recall are also always very close to 1 for this edge type after using the post-processing calibration method. These results indicate that when the calibrated probabilities indicate with high probability that there is no edge between a pair of variables, then those nodes rarely are directly causally related. This result provides confidence in not prioritizing the experimental investigation of such node pairs for direct causal relationships.

Another interesting observation in Figure 3 is that the bootstrap probabilities of the $A \leftrightarrow B$ edge-type are highly overestimated. This results in high false positive rate for that edge-type, and consequently, increases the false negative rate for other edge types. Note that post-processing the bootstrap probabilities does not generate high probabilities for the $A \leftrightarrow B$ edge-type and consequently there is no red circle in the high-probability bins but some in low probability bins, which is appropriate, because those edges, which are output by RFCI, are seldom correct.

Overall, the calibration diagrams in Figure 3 show that post-processing the bootstrap probabilities using the proposed shallow-neural network model generally improves the calibration performance of the predictions. A key advantage of the shallow neural network approach for post-processing the estimated probabilities is that we can readily condition on other types of features for learning a calibration mapping (e.g., features extracted from the structure of the predicted PAGs by the RFCI method, such as the indegree of B when we are generating a calibrated probability for the edge type $A \rightarrow B$). Such conditioning on local or global features of the learned graph could potentially yield improvements in the post-processed calibrated probabilities. This is an area for future research.

Table 1: Simulation results. V, E, and h represent the number of variables, edges, and percentage of hidden variables in the data-generating CBN, respectively. α is the significance level used in the RFCI method and N is the calibration training set size. Boldface indicates the results that are significantly better, based on the Wilcoxon signed rank test at 5% significance level. For MCE, lower is better.

(a) $V = 2000$, $h = 0.1$, and $\alpha = 0.001$

N	E	method	$A \to B$				$A \circ\!\!\to B$				$A \circ\!\!-\!\!\circ B$				$A \leftrightarrow B$				$A \cdots B$	Overall
			P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	MCE	MCE
70	2K	before	0.61	0.33	0.42	0.12	0.45	0.07	0.12	0.44	0.79	0.06	0.12	0.65	0.05	**0.25**	**0.08**	0.94	0.22	0.33
		after	**0.69**	**0.37**	**0.47**	0.10	**0.57**	**0.62**	**0.59**	**0.25**	0.79	**0.44**	**0.56**	**0.27**	0.03	0.02	0.02	**0.18**	**0.14**	**0.23**
	4K	before	0.66	0.30	0.41	0.30	**0.51**	0.03	0.06	0.45	0.62	0.03	0.05	0.44	**0.04**	**0.09**	**0.06**	0.95	0.51	0.55
		after	0.67	**0.38**	**0.48**	**0.20**	0.46	**0.43**	**0.44**	**0.23**	0.68	**0.27**	**0.38**	**0.26**	0.00	0.00	0.00	**0.09**	**0.21**	**0.22**
140	2K	before	0.57	0.28	0.37	0.10	0.42	0.06	0.10	0.46	0.64	0.05	0.09	0.65	0.04	**0.23**	**0.07**	0.93	0.21	0.34
		after	**0.66**	**0.31**	**0.41**	0.09	**0.58**	**0.63**	**0.60**	**0.27**	0.79	**0.41**	**0.54**	**0.29**	0.02	0.01	0.01	**0.17**	**0.12**	**0.26**
	4K	before	0.66	0.29	0.40	0.30	0.51	0.03	0.06	0.45	0.44	0.02	0.03	0.44	**0.04**	**0.09**	**0.06**	0.95	0.51	0.55
		after	**0.67**	**0.37**	**0.48**	**0.17**	0.47	**0.44**	**0.46**	**0.24**	**0.68**	**0.26**	**0.37**	**0.26**	0.00	0.00	0.00	**0.09**	**0.18**	**0.21**
210	2K	before	0.53	0.24	0.32	0.10	0.40	0.05	0.09	0.46	0.62	0.04	0.07	0.66	0.04	**0.23**	**0.07**	0.93	0.21	0.35
		after	**0.65**	**0.28**	**0.37**	0.09	**0.58**	**0.64**	**0.61**	**0.26**	0.80	**0.40**	**0.53**	**0.25**	0.02	0.01	0.01	**0.16**	**0.12**	**0.26**
	4K	before	0.65	0.29	0.40	0.31	0.51	0.03	0.05	0.46	0.27	0.01	0.02	0.44	**0.04**	**0.09**	**0.06**	0.95	0.51	0.56
		after	0.67	**0.36**	**0.47**	**0.19**	0.47	**0.43**	**0.45**	**0.23**	**0.66**	**0.23**	**0.34**	**0.29**	0.01	0.00	0.00	**0.08**	**0.17**	**0.20**

(b) $V = 2000$, $h = 0.2$, and $\alpha = 0.001$

N	E	method	$A \to B$				$A \circ\!\!\to B$				$A \circ\!\!-\!\!\circ B$				$A \leftrightarrow B$				$A \cdots B$	Overall
			P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	MCE	MCE
70	2K	before	0.35	0.15	0.21	0.10	0.37	0.03	0.05	0.48	0.56	0.03	0.05	0.66	0.11	**0.18**	0.13	0.90	0.24	0.34
		after	**0.43**	0.15	0.21	0.08	**0.53**	**0.34**	**0.41**	**0.26**	0.77	**0.30**	**0.42**	**0.27**	**0.21**	0.10	0.12	**0.29**	**0.13**	**0.24**
	4K	before	0.65	0.25	0.36	0.28	0.49	0.02	0.04	0.44	0.36	0.02	0.04	0.40	0.09	**0.06**	**0.07**	0.91	0.53	0.56
		after	0.63	**0.33**	**0.42**	**0.18**	0.44	**0.31**	**0.36**	**0.25**	**0.65**	**0.21**	**0.31**	**0.22**	0.08	0.01	0.01	**0.14**	**0.20**	**0.23**
140	2K	before	0.32	0.11	0.15	0.09	0.32	0.02	0.04	0.50	0.44	0.02	0.03	0.67	0.10	**0.17**	0.13	0.90	0.23	0.33
		after	0.39	0.10	0.14	0.08	**0.55**	**0.34**	**0.41**	**0.26**	0.79	**0.28**	**0.40**	**0.30**	**0.20**	0.10	0.11	**0.27**	**0.12**	**0.23**
	4K	before	0.65	0.24	0.35	0.28	0.48	0.02	0.03	0.45	0.20	0.01	0.03	0.40	0.09	**0.06**	**0.07**	0.91	0.53	0.59
		after	0.65	**0.31**	**0.42**	**0.15**	0.43	**0.31**	**0.36**	**0.24**	**0.65**	**0.22**	**0.32**	**0.20**	0.06	0.01	0.01	**0.14**	**0.19**	**0.21**
210	2K	before	0.26	0.09	0.13	0.10	0.32	0.02	0.03	0.50	0.40	0.01	0.02	0.67	0.10	**0.17**	0.13	0.90	0.23	0.33
		after	**0.35**	0.08	0.13	**0.08**	**0.55**	**0.32**	**0.40**	**0.27**	**0.80**	**0.27**	**0.39**	**0.31**	**0.19**	0.10	0.11	**0.28**	**0.12**	**0.23**
	4K	before	0.65	0.23	0.34	0.29	0.49	0.01	0.03	0.45	0.13	0.01	0.01	0.40	0.09	**0.06**	**0.07**	0.91	0.52	0.58
		after	0.65	**0.30**	**0.41**	**0.15**	0.43	**0.31**	**0.36**	**0.23**	**0.63**	**0.20**	**0.30**	**0.20**	0.14	0.01	0.01	**0.12**	**0.18**	**0.21**

(c) $V = 2000$, $h = 0.1$, and $\alpha = 0.005$

N	E	method	$A \to B$				$A \circ\!\!\to B$				$A \circ\!\!-\!\!\circ B$				$A \leftrightarrow B$				$A \cdots B$	Overall
			P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	MCE	MCE
70	2K	before	0.60	0.19	0.28	0.16	0.36	0.01	0.02	0.43	0.17	0.00	0.00	0.70	**0.05**	**0.30**	**0.09**	0.96	0.24	0.42
		after	0.65	**0.32**	**0.41**	**0.12**	0.43	**0.62**	**0.51**	**0.27**	**0.76**	**0.18**	**0.28**	**0.25**	0.00	0.00	0.00	**0.09**	**0.14**	**0.25**
	4K	before	**0.74**	0.29	0.41	0.29	0.39	0.01	0.01	0.46	0.18	0.00	0.00	0.46	0.05	**0.12**	**0.07**	0.97	0.38	0.55
		after	0.70	**0.39**	**0.50**	**0.20**	0.42	**0.44**	**0.43**	**0.27**	**0.66**	**0.14**	**0.22**	**0.28**	0.01	0.00	0.00	**0.10**	**0.17**	**0.24**
140	2K	before	0.49	0.13	0.20	0.16	0.27	0.01	0.01	0.43	0.00	0.00	0.00	0.69	**0.05**	**0.29**	**0.09**	0.96	0.24	0.38
		after	**0.62**	**0.23**	**0.32**	**0.11**	**0.44**	**0.62**	**0.51**	**0.26**	**0.77**	**0.18**	**0.28**	**0.24**	0.00	0.00	0.00	**0.09**	**0.13**	**0.24**
	4K	before	**0.74**	0.28	0.41	0.29	0.39	0.01	0.01	0.46	0.12	0.00	0.00	0.46	**0.05**	**0.11**	**0.06**	0.97	0.38	0.55
		after	0.70	**0.40**	**0.50**	**0.19**	0.44	**0.44**	**0.44**	**0.27**	**0.73**	**0.16**	**0.26**	**0.27**	0.00	0.00	0.00	**0.09**	**0.17**	**0.25**
210	2K	before	0.33	0.07	0.11	0.16	0.15	0.00	0.00	0.44	0.00	0.00	0.00	0.62	**0.05**	**0.29**	**0.09**	0.96	0.24	0.34
		after	0.43	**0.14**	**0.19**	**0.11**	**0.46**	**0.63**	**0.53**	**0.26**	**0.76**	**0.15**	**0.24**	**0.21**	0.00	0.00	0.00	**0.09**	**0.11**	**0.25**
	4K	before	**0.75**	0.27	0.40	0.30	0.35	0.00	0.01	0.47	0.10	0.00	0.00	0.47	**0.05**	**0.11**	**0.06**	0.97	0.38	0.54
		after	0.72	**0.37**	**0.48**	**0.18**	0.44	**0.45**	**0.44**	**0.26**	**0.73**	**0.17**	**0.27**	**0.25**	0.00	0.00	0.00	**0.08**	**0.15**	**0.24**

(d) $V = 2000$, $h = 0.2$, and $\alpha = 0.005$

N	E	method	$A \to B$				$A \circ\!\!\to B$				$A \circ\!\!-\!\!\circ B$				$A \leftrightarrow B$				$A \cdots B$	Overall
			P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	P	R	F1	MCE	MCE	MCE
70	2K	before	0.25	0.05	0.07	0.12	0.17	0.00	0.01	0.42	0.06	0.00	0.00	0.49	0.11	**0.21**	**0.15**	0.92	0.22	0.52
		after	0.39	**0.10**	**0.14**	0.09	**0.39**	**0.45**	**0.41**	**0.24**	**0.67**	**0.12**	**0.19**	**0.20**	0.06	0.01	0.02	**0.14**	**0.13**	**0.23**
	4K	before	**0.72**	0.22	0.34	0.30	0.32	0.00	0.01	0.45	0.12	0.00	0.00	0.43	0.10	**0.08**	**0.09**	0.94	0.40	0.49
		after	0.68	**0.30**	**0.42**	**0.22**	0.37	**0.32**	**0.35**	**0.27**	**0.70**	**0.12**	**0.20**	**0.25**	**0.05**	0.00	0.01	**0.13**	**0.18**	**0.25**
140	2K	before	0.15	0.02	0.03	0.11	0.14	0.00	0.00	0.44	0.00	0.00	0.00	0.47	0.11	**0.21**	**0.14**	0.92	0.22	0.50
		after	0.24	**0.06**	**0.09**	**0.07**	**0.40**	**0.45**	**0.42**	**0.25**	**0.72**	**0.11**	**0.17**	**0.20**	0.09	0.01	0.02	**0.14**	**0.12**	**0.23**
	4K	before	**0.73**	0.21	0.32	0.31	0.26	0.00	0.00	0.46	0.00	0.00	0.00	0.39	**0.10**	**0.08**	**0.08**	0.94	0.39	0.48
		after	0.71	**0.29**	**0.41**	**0.20**	0.38	**0.33**	**0.35**	**0.25**	**0.67**	**0.12**	**0.20**	**0.21**	0.04	0.00	0.01	**0.12**	**0.15**	**0.23**
210	2K	before	0.07	0.00	0.01	0.11	0.06	0.00	0.00	0.47	0.00	0.00	0.00	0.39	0.12	**0.20**	**0.15**	0.92	0.22	0.48
		after	0.13	0.02	0.03	**0.07**	**0.40**	**0.45**	**0.42**	**0.24**	**0.66**	**0.06**	**0.10**	**0.20**	0.09	0.01	0.02	**0.13**	**0.11**	**0.23**
	4K	before	**0.73**	0.20	0.31	0.32	0.26	0.00	0.00	0.46	0.00	0.00	0.00	0.37	**0.10**	**0.08**	**0.08**	0.94	0.40	0.48
		after	0.71	**0.28**	**0.40**	**0.21**	0.38	**0.33**	**0.35**	**0.26**	**0.71**	**0.12**	**0.20**	**0.21**	0.03	0.00	0.00	**0.12**	**0.16**	**0.25**

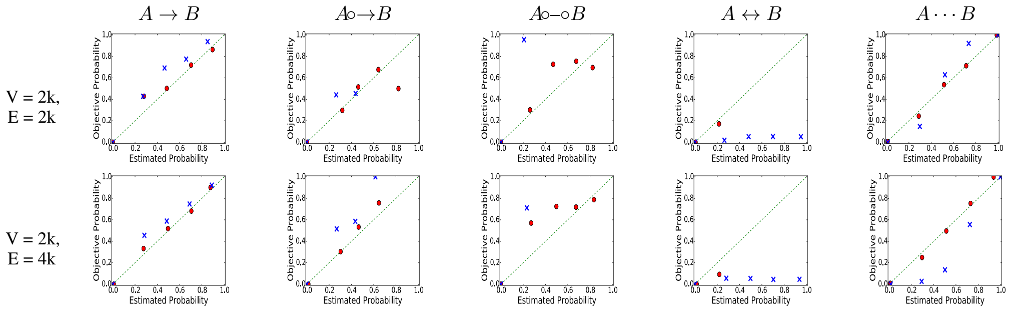

Figure 3: The calibration curves of the edge-type probabilities before (blue crosses) and after (red circles) calibration. The closer the predictions to the diagonal, the more calibrated are the probabilities. In these results, the calibration training set size is 210, the percentage of hidden variables is 0.1, and the significance level of the test of independence in RFCI is 0.005

5 Conclusion

In this paper, we introduced a new approach for improving the calibration of CBN structure discovery. We used a bootstrapping method to obtain estimated probabilities of the causal relationships between each pair of random variables. Although we applied the bootstrapping method to the RFCI algorithm, it can be applied with any other type of the network discovery method, as long as the method is sufficiently fast to run hundreds of times on a dataset to obtain bootstrap probability estimates. To calibrate the bootstrap probabilities, we devised a natural extension of Platt's calibration method that supports multi-class calibration using a shallow neural network. Our experiments on a wide range of large simulated datasets show that by using only a small set of instances as gold standards for training the calibration model, we can obtain substantial improvements in terms of precision, recall, and calibration, relative to the bootstrap probabilities. In future work, we plan to expand the range of simulated experiments we perform, as well as evaluate the method using real biomedical data for which the truth status is known from the literature for a relatively small subset of variables.

Acknowledgement We thank the members of the Center for Causal Discovery for their helpful feedback. The research reported in this publication was supported by grant U54HG008540 awarded by the National Human Genome Research Institute through funds provided by the trans-NIH Big Data to Knowledge (BD2K) initiative. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. This material is also based upon work supported by the National Science Foundation under Grant No. IIS-1636786. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] Martín Abadi, Ashish Agarwal, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv:1603.04467*, 2016.

[2] David Maxwell Chickering. Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3:507–554, 2003.

[3] Diego Colombo, Marloes Maathuis, Markus Kalisch, and Thomas Richardson. Learning high-dimensional directed acyclic graphs with latent and selection variables. *The Annals of Statistics*, 40(1):294–321, 2012.

[4] Morris DeGroot and Stephen Fienberg. The comparison and evaluation of forecasters. *The Statistician*, 1983.

[5] Pedro Domingos. A few useful things to know about machine learning. *Communications of the ACM*, 55(10):78–87, 2012.

[6] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[7] Daniel Eaton and Kevin Murphy. Exact Bayesian structure learning from uncertain interventions. In *International Conference on Artificial Intelligence and Statistics*, pages 107–114, 2007.

[8] Bradley Efron and Robert Tibshirani. *An Introduction to the Bootstrap*. CRC press, 1994.

[9] Nir Friedman, Moises Goldszmidt, and Abraham Wyner. Data analysis with Bayesian networks: A bootstrap approach. In *Proceedings of the Fifteenth Conference on Uncertainty in artificial intelligence*, pages 196–205, 1999.

[10] Nir Friedman and Daphne Koller. Being Bayesian about network structure. a Bayesian approach to structure discovery in Bayesian networks. *Machine learning*, 50(1-2):95–125, 2003.

[11] Clark Glymour and Gregory Cooper. *Computation, Causation, and Discovery*. MIT Press, 1999.

[12] Phyllis Illari, Federica Russo, and Jon Williamson. *Causality in the Sciences*. OUP Oxford, 2011.

[13] Mikko Koivisto. Advances in exact Bayesian structure discovery in Bayesian networks. *arXiv:1206.6828*, 2012.

[14] Mikko Koivisto and Kismat Sood. Exact Bayesian structure discovery in Bayesian networks. *Journal of Machine Learning Research*, 5:549–573, 2004.

[15] David Madigan, Jeremy York, and Denis Allard. Bayesian graphical models for discrete data. *International Statistical Review*, pages 215–232, 1995.

[16] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Binary classifier calibration using a Bayesian non-parametric approach. In *Proceedings of the SIAM International Conference on Data Mining*, pages 208–216, 2015.

[17] Mahdi Pakdaman Naeini, Gregory Cooper, and Milos Hauskrecht. Obtaining well calibrated probabilities using Bayesian binning. In *AAAI*, pages 2901–2907, 2015.

[18] Michael A Nielsen. Neural networks and deep learning. *Determination Press*, 2015.

[19] Judea Pearl. Causality: Models, reasoning and inference. *Econometric Theory*, 19:675–685, 2003.

[20] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. *Advances in Large Margin Classifiers*, 10(3):61–74, 1999.

[21] Joseph Ramsey. Scaling up greedy equivalence search for continuous variables. *arXiv:1507.07749*, 2015.

[22] Ricardo Silva, Richard Scheines, Clark Glymour, and Peter Spirtes. Learning the structure of linear latent variable models. *Journal of Machine Learning Research*, 7:191–246, 2006.

[23] Peter Spirtes. Introduction to causal inference. *Journal of Machine Learning Research*, 11:1643–1662, 2010.

[24] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT press, 2000.

[25] Bianca Zadrozny and Charles Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 694–699, 2002.