Causal Feature Selection With Dual Correction

Xianjie Guo, Kui Yu, *Member, IEEE*, Lin Liu, Fuyuan Cao, and Jiuyong Li, *Member, IEEE*

***Abstract*— Causal feature selection methods aim to identify a Markov boundary (MB) of a class variable, and almost all the existing causal feature selection algorithms use conditional independence (CI) tests to learn the MB. However, in real-world applications, due to data issues (e.g., noisy or small samples), CI tests can be unreliable; thus, causal feature selection algorithms relying on CI tests encounter two types of errors: false positives (i.e., selecting false MB features) and false negatives (i.e., discarding true MB features). Existing algorithms only tackle either false positives or false negatives, and they cannot deal with both types of errors at the same time, leading to unsatisfactory results. To address this issue, we propose a dual-correction-strategy-based MB learning (DCMB) algorithm to correct the two types of errors simultaneously. Specifically, DCMB selectively removes false positives from the MB features currently selected, while selectively retrieving false negatives from the features currently discarded. To automatically determine the optimal number of selected features for the selective removal and retrieval in the dual correction strategy, we design the simulated-annealing-based DCMB (SA-DCMB) algorithm. Using benchmark Bayesian network (BN) datasets, the experimental results demonstrate that DCMB achieves substantial improvements on the MB learning accuracy compared with the existing MB learning methods. Empirical studies in real-world datasets validate the effectiveness of SA-DCMB for classification against state-of-the-art causal and traditional feature selection algorithms.**

***Index Terms*— Bayesian network (BN), dual correction, feature selection, Markov boundary (MB).**

I. Introduction

THE Markov boundary (MB) is a key concept in a Bayesian network (BN). If the probability distribution of a dataset can be faithfully represented by a BN, the MB of a node (feature) in the BN consists of its parents (direct causes), children (direct effects), and spouses (other parents of the children) of the node [1].

The MB of a variable provides a complete picture of the local causal relationships around the variable, and thus, learning MBs plays an essential role in scalable learning of both local and global causal structures [2], [3]. Moreover, it has been proven that MB of the class variable is theoretically the minimal feature subset of optimal feature selection for prediction when data distribution is faithfully represented by a BN [4]–[6].

Accordingly, as an emerging successful filtering feature selection approach, causal feature selection [7] has attracted much attention inspired by the theory of BN and MB. Causal feature selection aims to learn the MB of a class variable of the dataset, and many causal feature selection algorithms have been developed in the past decade [8]. Almost all these algorithms use conditional independence (CI) tests to assess the associations between features as a statistical measure to identify the MB of the class variable.

However, in real-world applications, limited by small sample size, noisy data, or high dimensionality, CI tests can be unreliable during MB learning, leading to two types of errors of the CI test-based causal feature selection methods: false positives (i.e., some false MB features being selected) and false negatives (i.e., some true MB features being discarded). To rectify the two types of errors, the existing causal feature selection algorithms use either the AND rule to remove false positives (e.g., parents and children based MB (PCMB) [9] and Iterative parent-Child based search of MB (IPCMB) [10]) or the OR rule to retrieve false negatives (e.g., cross-check and complement MB discovery (CCMB) [11]).

Moreover, we use the example in Fig. 1 to illustrate how the existing algorithms use either the AND rule or the OR rule to resolve the two types of errors and summarize their shortcomings. In this example, we choose the frequently used ALARM BN and generate two synthetic datasets (1000 data samples each). Using the ALARM network, we can read off the true MB of a node, and then we are able to compare the MB of the node learned from a dataset with the true MB of the node in the network. In Fig. 1(a), the true MB of X_{15} in the ALARM network is highlighted in orange. A red cross denotes that a true MB feature is discarded by an algorithm, and a red arrow represents that a false MB feature is selected by an algorithm.

We choose two well-established causal feature selection algorithms, PCMB [9] and max-min MB (MMMB) [12], and one state-of-the-art algorithm, CCMB [11]. PCMB uses the AND rule to remove false positives, while CCMB employs the OR rule to retrieve false negative. MMMB does not use either rule.

In a BN, the AND rule means that if feature F_i is a parent of feature F_j (i.e., $F_i \rightarrow F_j$), then F_j must be a child of F_i. However, in real-world applications, due to noisy data, it was often found that when using a causal feature selection

Manuscript received June 12, 2021; revised November 23, 2021 and March 12, 2022; accepted May 15, 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2020AAA0106100, in part by the National Natural Science Foundation of China under Grant 61876206, and in part by the Australian Research Council Discovery Projects under Grant DP200101210. *(Corresponding author: Kui Yu.)*

Xianjie Guo and Kui Yu are with the Intelligent Interconnected Systems Laboratory of Anhui Province and the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei 230601, China (e-mail: xianjieguo@mail.hfut.edu.cn; yukui@hfut.edu.cn).

Lin Liu and Jiuyong Li are with UniSA STEM, University of South Australia, Adelaide, SA 5095, Australia (e-mail: lin.liu@unisa.edu.au; jiuyong.li@unisa.edu.au).

Fuyuan Cao is with the School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China (e-mail: cfy@sxu.edu.cn).

This article has supplementary material provided by the authors and color versions of one or more figures available at https://doi.org/10.1109/TNNLS.2022.3178075.

Digital Object Identifier 10.1109/TNNLS.2022.3178075

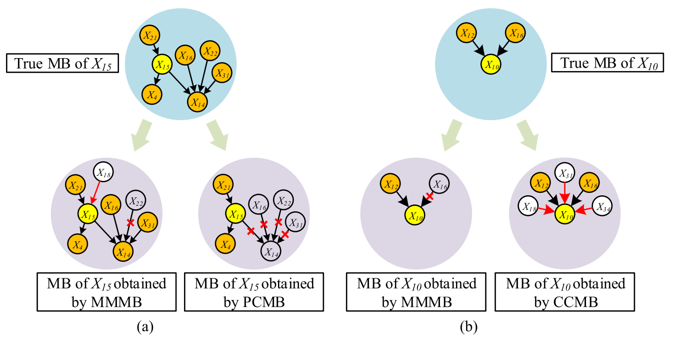
True MB of X_{15}
True MB of X_{10}
MB of X_{15} obtained by MMMB
MB of X_{15} obtained by PCMB
MB of X_{10} obtained by MMMB
MB of X_{10} obtained by CCMB
(a)
(b)

Fig. 1. (a) MBs learned by MMMB with PCMB using 1000 samples. (b) MBs learned by MMMB with CCMB using 1000 samples.

algorithm to find the parents and children (PC) set of F_i, F_j was not found in the PC set of F_i; however, the found PC set of F_j included F_i. Thus, the OR rule means that if F_j is not found in the PC set of F_i but F_i is found in the PC set of F_j, F_j is still considered a true PC feature of F_i.

For X_{15}, PCMB first learns the PC set of X_{15} and then learns *the PC set of each feature (denoted as A) within the learned PC set of* X_{15}. Using the AND rule, if X_{15} is not in the learned PC set of A, A is considered a false positive and is removed from the learned PC set of X_{15}. After removing false PC features, PCMB learns the spouses of X_{15}. In Fig. 1(a), compared with the MB learned by MMMB, PCMB removes false positives (e.g., X_{18}), as well as many true MB features (e.g., X_{14}, X_{16}, and X_{31}). In Fig. 1(b), CCMB also first learns the PC set of X_{10}; then, it learns *the PC set of each feature (denoted as B) within all discarded features*. Clearly, B is not in the learned PC set of X_{10}. Using the OR rule, if the found PC set of B includes X_{10}, B is considered a false negative, and it is added to the learned PC set of X_{10}. After retrieving the missed true PC features, CCMB learns the spouses of X_{10}. From Fig. 1(b), compared with the MB learned by MMMB, CCMB not only recovers the true MB feature X_{16} but also selects three false MB features, such as X_{18}, X_{31}, and X_{34}.

Compared with the MMMB algorithm, in Fig. 1, it was found that: 1) PCMB and CCMB may not obtain satisfactory results and 2) PCMB and CCMB are more computationally expensive than MMMB since using the AND rule, PCMB needs to perform additional CI tests for finding the PC sets of all features within the found PC set; while using the OR rule, CCMB needs to conduct additional CI tests for learning the PC sets of all the discarded features.

The above findings lead to the following question: can we use both the rules at the same time to simultaneously reduce both false positive and false negative errors to achieve satisfactory results with relatively low computational costs?

This article tackles the question raised above and makes the following main contributions.

1) We propose the dual-correction-strategy-based MB learning (DCMB) algorithm. In the DCMB algorithm, we design a dual correction strategy using the AND rule and the OR rule together to simultaneously reduce both false positive and false negative errors. To make both the rules work efficiently and effectively, for the AND rule, instead of using the entire found PC set, the dual correction strategy selectively chooses features within the found PC set for finding their PC sets to remove false positives. For the OR rule, rather than using all the discarded features, this strategy selectively chooses features within the discarded features for learning their PC sets to retrieve false negatives.
2) To automatically search for the optimal number of selected features for the selective removal and retrieval in the dual correction strategy, we design the simulated-annealing-based DCMB (SA-DCMB) algorithm, which uses simulated annealing (SA) to automatically determine the optimal number of selective features for both the rules.
3) Using synthetic datasets, we validate the quality of the learned MB and the learning efficiency of DCMB. Using real-world datasets, we validate the effectiveness of SA-DCMB for classification against the well-established and state-of-the-art feature selection algorithms.

II. Related Work

Feature selection has been widely studied and used in machine learning, as well as in data mining. In addition, many excellent studies on feature selection methods have been presented recently. We suggest readers refer to reviews on this topic, such as [13] for an extensive survey on classic feature selection methods, [14] and [15] for a review of information theoretic feature selection algorithms and [8] for a comprehensive survey on causal feature selection methods.

In this section, we focus on causal feature selection methods. The existing causal feature selection methods can be divided into multi-source and single-source causal feature selection methods. The former selects causal features from multiple datasets with different distributions [16], while the latter learns causal features using a single dataset [17]. Recently, Mastakouri *et al.* [18] proposed a novel and sound causal feature selection algorithm to deal with time series data with latent variables. In this article, we focus on learning causal features from a single dataset in a static environment.

We first briefly introduce the causal feature selection algorithms, and then highlight the causal feature selection methods using the AND rule or the OR rule to correct false positive or false negative errors. Koller and Sahami [17] were the first to introduce MB to feature selection. Based on this work [17], in the past decade, many causal feature selection algorithms have been developed [8] without learning an entire BN, and thus were able to scale up to hundreds of thousands of features. In general, the existing causal feature selection algorithms using a single dataset can be divided into two main types: simultaneous MB (STMB) learning and divide-and-conquer MB learning. A STMB learning algorithm finds parents, children, and spouses of the class variable simultaneously and does not distinguish the PC of the class variable from its spouses during MB learning. The grow-shrink MB (GSMB) algorithm was the first correct algorithm for learning an MB of the class variable [19]. Based on GSMB, many of its variants have been developed, such as incremental association MB (IAMB) [5], Fast-IAMB [20], Inter-IAMB [21], IAMB-informative prior (IAMB-IP) [22], and forward-backward selection with early dropping $(\text{FBED})^K$ [23]. Since those MB learning algorithms use the entire currently selected MB as the conditioning set at each computation, the data samples required by those algorithms are exponential to the size of the MB set that is currently selected. Moreover, the existing STMB learning algorithms do not use the AND rule or the OR rule.

The divide-and-conquer MB learning methods were proposed to reduce the data requirements of the STMB learning methods. This approach breaks down the MB learning problem into two subproblems: PC learning (i.e., learning PC of the class variable) and spouse learning (i.e., learning spouses of the class variable). The representative algorithms include: MMMB [12], HITON-MB [24], PCMB [9], [25], MB search using the OR condition (MBOR) [26], IPCMB [10], STMB [27], CCMB [11], and separation and recovery MB discovery (SRMB) [28]. To achieve a balance between data efficiency and time efficiency, balanced MB discovery (BAMB) [29] and efficient and effective discovery of MB (EEMB) [30] perform the PC learning step and the spouse discovery step alternatively instead of performing these two steps separately. For PC learning, these methods do not use the entire PC set currently selected as the conditioning set for CI tests. Instead, it makes use of the subsets of the PC set currently selected, which is much smaller than the entire MB set used by the STMB learning approach when determining dependence/independence relationships between variables. Thus, the divide-and-conquer MB learning approach requires a considerably smaller number of samples than the STMB learning approach.

Since both the AND and OR rules use the symmetry property of PC in BNs, the existing algorithms using either rule focus on the type of divide-and-conquer MB learning methods. Peña *et al.* [25] and Tsamardinos *et al.* [31] independently claimed that the MMMB and HITON-MB algorithms cannot return the correct MB even under the faithfulness assumption. They found that both the MMPC and the HITON-PC algorithms used by MMMB and HITON-MB, respectively, may return a superset of the true PC of C (i.e., including false positives). Then, Pena *et al.* [9] proposed the PCMB algorithm. Based on the idea of PCMB, the IPCMB algorithm [10] was proposed. Both the algorithms use the AND rule to remove false positives. Specifically, for each feature in the found PC set of the class variable, if the PC set of the feature does not include the class variable, then the two algorithms consider the feature as a false positive and remove it from the found PC set.

However, as shown in Fig. 1, the current algorithms using the AND rule remove not only false positives but also true positives. That is to say, using the AND rule reduces the false positive rate, but it increases the false negative rate. Then, to reduce the false negative rate, the MBOR algorithm was designed using the OR rule. Clearly, the OR rule is less strict than the AND rule and makes it easier for true positives to enter the MB. Since MBOR uses the STMB discovery approach for PC learning, it still struggles with the problem of data inefficiency. To tackle this problem, Wu *et al.* [11], [28] proposed the CCMB algorithm and the SRMB algorithm. The CCMB algorithm adopts the divide-and-conquer idea to learn PC of the class variable for alleviating the data-efficient problem. Then, it adopts the OR rule to recover the discarded true positives. SRMB first learns a rough MB of C by a fast MB discovery algorithm, then proceeds with a separation process to separate PC and spouses from the initial MB, and finally exploits the OR rule to recover the missed PC and spouses of C.

However, the current algorithms using the OR rule can recover as many true positives as possible, but they increase the number of false positives. Moreover, MBOR, CCMB and SRMB all need to identify the PC sets of all features excluding the found PC set, which leads to very expensive computations. In this article, we investigate the OR rule and the AND rule and study how to make both the rules truly work in practice.

III. Notations and Definitions

In this section, we provide the basic notations and definitions related to causal feature selection, BN, and MB used throughout the article. We let $\boldsymbol{F}$ consist of M features and the class variable C, i.e., $\boldsymbol{F} = \{F_1, F_2, \ldots, F_{M+1}\}$, where $F_i = F_i$ $(1 \le i \le M)$, and $F_{M+1} = C$. Let $\boldsymbol{S}$ be any set of features within $\boldsymbol{F}\backslash\{C\}$, and we use $\boldsymbol{S}\backslash F_i$ as the shorthand of $\boldsymbol{S}\backslash\{F_i\}$ and $\boldsymbol{S} \cup F_i$ as the shorthand of $\boldsymbol{S} \cup \{F_i\}$. We use $F_i \perp\!\!\!\perp F_j|\boldsymbol{S}$ to denote that F_i and F_j are conditionally independent given $\boldsymbol{S} \subseteq \boldsymbol{F}\backslash\{F_i, F_j, C\}$, and $F_i \not\!\perp\!\!\!\perp F_j|\boldsymbol{S}$ to represent that F_i and F_j are conditionally dependent given $\boldsymbol{S}$. The symbols $\boldsymbol{PC}(F_i)$ and $\boldsymbol{SP}(F_i)$ denote the set of PC and the set of spouses of F_i, respectively.

Let $\mathbb{P}(\boldsymbol{F})$ be the joint probability distribution over $\boldsymbol{F}$ and $\mathbb{G} = (\boldsymbol{F}, \boldsymbol{E})$ represents a directed acyclic graph (DAG) with nodes $\boldsymbol{F}$ and edges $\boldsymbol{E}$, where an edge $F_i \to F_j$ denotes that F_i is a parent of F_j while F_j is a child of F_i. The triplet $\langle \boldsymbol{F}, \mathbb{G}, \mathbb{P}(\boldsymbol{F})\rangle$ is called a BN if and only if $\langle \boldsymbol{F}, \mathbb{G}, \mathbb{P}(\boldsymbol{F})\rangle$ satisfies the Markov condition: every node of $\mathbb{G}$ is independent of any subset of its non-descendants conditioning on the parents of the node [1].

In a BN, due to the symmetry relationship of a node and its parents (or its children), the AND rule and the OR rule are defined as follows. Clearly, the OR rule is less strict than the AND rule.

*Definition 1 (*AND *rule): In a BN, if F_i is a parent of F_j [i.e., $F_i \in \boldsymbol{PC}(F_j)$], F_j must be a child of F_i [i.e., $F_j \in \boldsymbol{PC}(F_i)$].*

*Definition 2 (*OR *rule): Given a causal feature selection algorithm and a dataset for learning the PC set of F_i, if F_j is not in the learned PC set of F_i [i.e., $F_j \notin \boldsymbol{PC}(F_i)$] but F_i is in the learned PC set of F_j [i.e., $F_i \in \boldsymbol{PC}(F_j)$], F_j is still considered a true PC feature of F_i.*

Definition 3 (Faithfulness [1]): Given a BN $\langle \boldsymbol{F}, \mathbb{G}, \mathbb{P}(\boldsymbol{F})\rangle$, $\mathbb{G}$ is faithful to $\mathbb{P}(\boldsymbol{F})$ if and only if every CI present in $\mathbb{P}$ is entailed by $\mathbb{G}$ and the Markov condition. $\mathbb{P}(\boldsymbol{F})$ is faithful to $\mathbb{G}$ if and only if there exists a DAG $\mathbb{G}$ such that $\mathbb{G}$ is faithful to $\mathbb{P}(\boldsymbol{F})$.

Under the faithfulness assumption, Definition 4 below states the uniqueness of MBs and what the MB of a node is in a BN.

Definition 4: (MB [1]) Under the faithfulness assumption, the MB of any node in a BN is unique and it consists of the node's parents, children, and spouses (other parents of the node's children).

In BNs, the MB of a node renders the node statistically independent of all the remaining nodes conditioning on the MB [1], as shown in Proposition 1 below.

Proposition 1 ([1]): In a BN, let $\boldsymbol{MB}(F_i)$ be the MB of node F_i, $\forall F_j \in \boldsymbol{F} \setminus (\boldsymbol{MB}(F_i) \cup C)$, $F_i \perp\!\!\!\perp F_j | \boldsymbol{MB}(F_i)$ holds.

Proposition 1 bridges the gap between MB learning and feature selection and illustrates that learning the MB of the class variable is actually a procedure of optimal feature selection.

Proposition 2 ([5], [6]): Under the faithfulness assumption, $\forall F_i \in \boldsymbol{F}$, F_i belongs to the MB of the class variable C ($\boldsymbol{MB}(C)$), if and only if F_i is a strongly relevant feature.

IV. Proposed DCMB Algorithm

In this section, we propose the DCMB algorithm, and this section is organized as follows. We first give the overview of the DCMB algorithm in Section IV-A, and then present the detailed description of DCMB in Section IV-B.

A. Overview of DCMB

The DCMB algorithm includes four phases as shown in Algorithm 1.

For Phase I (Line 1), we propose the IdenCPC algorithm (presented in Algorithm 2) to identify candidate PC (IdenCPC) of C ($\boldsymbol{CPC}$).

The dual correction is done in Phase II (Line 2) and Phase III (Line 3) and these two phases are the most critical parts of DCMB. For Phase II and Phase III, we propose the OR rule-based PC retrieval (ORPC) and AND rule-based PC removal (ANDPC) algorithms, respectively (presented in Algorithms 3 and 4). The ORPC algorithm uses the OR rule to recover the true PC discarded in Phase I, while the ANDPC algorithm use the AND rule to remove false positives from $\boldsymbol{CPC}$. To efficiently and effectively apply the two rules, the ORPC and ANDPC algorithms use two input parameters k_or $\in [0, 1]$ (denoting any real number between 0 and 1) and k_and $\in [0, 1]$ to greedily select some features for correcting the two types of errors, respectively, and these selected features have a high probability of being identified as false negatives or false positives.

Finally, Phase IV (Lines 5–12) learns the spouses of the class variable (denoted as $\boldsymbol{SP}$) based on the corrected PC set in Line 4. In this phase, the idea of identifying spouses is the same as the existing MB learning algorithms, such as MMMB.

B. Detailed Descriptions of DCMB

1) IdenCPC Algorithm in Phase I: The IdenCPC algorithm adopts the similar idea of the MMPC algorithm [12] to learn a candidate PC set of the class variable, denoted as $\boldsymbol{CPC}$. The pseudo code of IdenCPC is shown in Algorithm 2, which consists of two steps: the forward step (Step 1) and the backward step (Step 2).

Step 1 (Forward step, Lines 2–15 in Algorithm 2.) We assume that both $\boldsymbol{CPC}$ and $\boldsymbol{or\_rank}$ are initially empty. $\boldsymbol{CPC}$ stores the candidate PC of C found by IdenCPC, while $\boldsymbol{or\_rank}$ keeps possibly discarded true PC for recovering later using the OR rule.

Algorithm 1 DCMB

```
Input: $C$: the class variable; $k\_or \in [0,1]$; $k\_and \in [0,1]$
Output: MB of $C$
    {Phase I: IdenCPC}
 1: $[\boldsymbol{or\_rank}, \boldsymbol{CPC}] = IdenCPC(C)$
    {Phase II: The "OR" rule for recovering discarded PC}
 2: $\boldsymbol{orPC} = ORPC(k\_or, \boldsymbol{or\_rank})$
    {Phase III: The "AND" rule for removing false PC}
 3: $\boldsymbol{andCPC} = ANDPC(k\_and, \boldsymbol{CPC})$
 4: $\boldsymbol{PC} = \boldsymbol{andCPC} \cup \boldsymbol{orPC}$
    {Phase IV: Find spouses}
 5: $\boldsymbol{SP} = \emptyset$
 6: for each $X \in \boldsymbol{PC}$ do
 7:   for each $Y \in \boldsymbol{PC}(X)$ and $Y \notin \boldsymbol{PC}$ do
 8:     if $\exists\, \boldsymbol{S}$ s.t. $C \perp\!\!\!\perp Y|\boldsymbol{S}$ and $C \not\perp\!\!\!\perp Y|\boldsymbol{S} \cup \{X\}$ then
 9:         $\boldsymbol{SP} \longleftarrow \boldsymbol{SP} \cup \{Y\}$
10:     end if
11:   end for
12: end for
13: $\boldsymbol{MB} = \boldsymbol{PC} \cup \boldsymbol{SP}$
```

Given the $\boldsymbol{CPC}$ currently selected, for each feature $X \in \boldsymbol{F} \setminus \boldsymbol{CPC}$, in Line 4, IdenCPC uses the function dep() to calculate the relevancy of X and C conditioning on all possible subsets of $\boldsymbol{CPC}$, and then chooses the minimum relevancy as the relevancy value of X and C. The function dep() can be instantiated by chi-squared test, mutual information, etc. Dep[X] keeps the minimum relevancy value of X and C conditioning on the subset Sep[X] within $\boldsymbol{CPC}$. In Line 5, if X is independent of C, X is removed from $\boldsymbol{F}$ and never considered as candidate PC again in Line 6.

The smaller the size of $\boldsymbol{or\_rank}$, the more efficient the ORPC algorithm. In Lines 7 and 8, to keep $\boldsymbol{or\_rank}$ as small as possible, we only consider that if X and C are independent and their conditioning set is not empty, X is added to the set $\boldsymbol{or\_rank}$ at Line 8. The explanation for this is that if X is independent of C conditioning on an empty set, both $C \notin \boldsymbol{PC}(X)$ and $X \notin \boldsymbol{PC}(C)$ hold and thus X cannot be recovered using the OR rule.

Then in Lines 12 and 13, IdenCPC selects the next feature to be included in $\boldsymbol{CPC}$ as the one that is dependent of C and exhibits the maximum relevancy among the features in $\boldsymbol{F} \setminus \boldsymbol{CPC}$. The forward step is terminated until $\boldsymbol{F}$ is empty (that is to say, each feature in $\boldsymbol{F} \setminus \boldsymbol{CPC}$ and C are independent given any subsets of $\boldsymbol{CPC}$).

Step 2 (Backward step, Lines 16–21 in Algorithm 2.) At the backward phase, IdenCPC checks whether each feature Y in $\boldsymbol{CPC}$ is independent of C conditioning on all possible subsets of $\boldsymbol{CPC} \setminus Y$. If so, Y is removed from $\boldsymbol{CPC}$ and Y is added to $\boldsymbol{or\_rank}$; otherwise, it is retained.

2) ORPC Algorithm in Phase II: The ORPC algorithm is shown in Algorithm 3. The basic idea of the ORPC algorithm is to determine which features in $\boldsymbol{or\_rank}$ are the discarded PC of C. If the PC set of each feature $X \in \boldsymbol{or\_rank}$ includes

Algorithm 2 IdenCPC

```
Input: $C$; $\boldsymbol{F}$: union of features and class variable
Output: $\boldsymbol{or\_rank}$: possibly discarded true positives;
        $\boldsymbol{CPC}$: candidate parents and children features
1: Initialize $\boldsymbol{or\_rank} = \emptyset$, $\boldsymbol{CPC} = \emptyset$, $\boldsymbol{F} = \boldsymbol{F} \setminus \{C\}$
   {Step 1: Forward step}
2: repeat
3:   for each $X \in \boldsymbol{F}$ do
4:     $[Dep[X], Sep[X]] = \arg\min_{\boldsymbol{S} \subseteq \boldsymbol{CPC}} dep(C, X|\boldsymbol{S})$
5:     if $C \perp\!\!\!\perp X|Sep[X]$ then
6:       $\boldsymbol{F} = \boldsymbol{F} \setminus \{X\}$
7:       if $Sep[X] \neq \emptyset$ then
8:         $\boldsymbol{or\_rank} \longleftarrow \boldsymbol{or\_rank} \cup \{X\}$
9:       end if
10:    end if
11:  end for
12:  $Y = \arg\max_{X \in \boldsymbol{F}} Dep(X)$
13:  $\boldsymbol{CPC} = \boldsymbol{CPC} \cup \{Y\}$
14:  $\boldsymbol{F} = \boldsymbol{F} \setminus \{Y\}$
15: until $\boldsymbol{F} = \emptyset$
    {Step 2: Backward step}
16: for each $X \in \boldsymbol{CPC}$ do
17:   if $\exists \boldsymbol{S} \subseteq \boldsymbol{CPC} \setminus \{X\}$ such that $C \perp\!\!\!\perp X|\boldsymbol{S}$ then
18:     $\boldsymbol{CPC} = \boldsymbol{CPC} \setminus \{X\}$
19:     $\boldsymbol{or\_rank} \longleftarrow \boldsymbol{or\_rank} \cup \{X\}$
20:   end if
21: end for
```

Algorithm 3 ORPC

```
Input: $k\_or \in [0,1]$; $\boldsymbol{or\_rank}$
Output: $\boldsymbol{orPC}$: recovered PC by the "OR" rule
1: Initialize $\boldsymbol{orPC} = \emptyset$
   /*Descending order, $F_1$ has the highest dependency*/
2: $\langle F_1, \ldots, F_{|\boldsymbol{or\_rank}|} \rangle \longleftarrow \boldsymbol{or\_rank}$
3: for i = 1 to $R(|\boldsymbol{or\_rank}| \times k\_or)$ do
4:   $[\boldsymbol{or\_rank2}, \boldsymbol{CPC2}] = IdenPC(F_i)$
5:   if $C \in \boldsymbol{CPC2}$ then
6:     $\boldsymbol{orPC} = \boldsymbol{orPC} \cup \{F_i\}$
7:   end if
8: end for
```

C, X is reconsidered as a parent or a child of C and added to $\boldsymbol{orPC}$. First, it is computationally expensive to check all the features in $\boldsymbol{or\_rank}$, especially when the size of $\boldsymbol{or\_rank}$ becomes large. Second, due to incorrect CI tests, more features within $\boldsymbol{or\_rank}$ are examined, and more false positives may be added. To save computational costs and ensure that only a few false positives are added, ORPC adopts a selective correction strategy that uses a parameter k_or $\in [0, 1]$ to control the number of features selected from $\boldsymbol{or\_rank}$.

To implement this strategy, in Line 2, ORPC first sorts the features in $\boldsymbol{or\_rank}$ in descending order according to the association values between these features and C. Second, ORPC uses the parameter k_or to specify the number of features selected from $\boldsymbol{or\_rank}$. Then, in Lines 3 and 4, ORPC examines the $R(|\boldsymbol{or\_rank}| \times k$_or) features from the first feature in $\boldsymbol{or\_rank}$ and identifies their PC set using the IdenCPC algorithm. $|\boldsymbol{or\_rank}|$ denotes the number of features in $\boldsymbol{or\_rank}$ and the function $R(m)$ is used to round the parameter m. In Lines 5 and 6, if a feature's PC set includes C, then the feature is added to $\boldsymbol{orPC}$. For the parameter k_or, if k_or = 1, ORPC will identify the PC sets of all the features in $\boldsymbol{or\_rank}$, while if taking k_or = 0, ORPC does not examine any features in $\boldsymbol{or\_rank}$. In addition, we sort the features in $\boldsymbol{or\_rank}$ in descending order since the higher dependency a feature has with C, the higher chance the feature has to be a true positive. This strategy makes ORPC selectively choose the most promising features in $\boldsymbol{or\_rank}$ for recovering, while avoiding checking all the features in $\boldsymbol{or\_rank}$. We discuss the empirical bound of the value of k_or in Section VI-B.

3) ANDPC Algorithm in Phase III: To remove false positives from $\boldsymbol{CPC}$ achieved in Phase I, th existing algorithms use the AND rule to examine all the features in $\boldsymbol{CPC}$. However, examining all the features in $\boldsymbol{CPC}$ is not only computationally expensive but also removes true positives from $\boldsymbol{CPC}$.

The ANDPC algorithm (as shown in Algorithm 4) also uses the AND rule, but it implements a selective correction by specifying a parameter k_and $\in [0, 1]$ to control the number of features selected from $\boldsymbol{CPC}$. Specifically, in Line 2, ANDPC first sorts the features in $\boldsymbol{CPC}$ in ascending order according to the association values between these features and C. In Line 3, ANDPC specifies k_and to control the number of features selected from $\boldsymbol{CPC}$. In Line 4, ANDPC examines the $R(|\boldsymbol{CPC}| \times k$_and) features from the first feature in $\boldsymbol{CPC}$ and identifies their PC sets using IdenCPC. $|\boldsymbol{CPC}|$ denotes the number of features in $\boldsymbol{CPC}$ and $R(m)$ is the same as $R(m)$ in the ORPC algorithm. In Lines 5 and 6, if a feature's PC set does not include C, then the feature is removed from $\boldsymbol{andCPC}$.

For the parameter k_and, if k_and = 1, ANDPC learns the PC sets of all the features in $\boldsymbol{CPC}$, while if taking k_and = 0, ANDPC does not check any features in $\boldsymbol{CPC}$. In addition, the features in $\boldsymbol{CPC}$ are arranged in ascending order. The explanation is that if a feature has a low dependency with C, the feature has a high chance to be a false positive. This strategy makes ANDPC selectively choose the most promising features for being removed from $\boldsymbol{CPC}$ and at the same time avoid checking all the features in $\boldsymbol{CPC}$. We will discuss the empirical bound of the value of k_and in Section VI-B.

The order of Lines 2 and 3 in Algorithm 1 has no effect on the result of DCMB. According to the IdenCPC algorithm (see Algorithm 2), $\boldsymbol{or\_rank} \cap \boldsymbol{CPC} = \emptyset$ holds. Here, both $\boldsymbol{or\_rank}$ and $\boldsymbol{CPC}$ are the outputs of IdenCPC. $\boldsymbol{orPC} \cap \boldsymbol{andCPC} = \emptyset$ also holds since the $\boldsymbol{orPC}$ output by the ORPC algorithm in Phase II is a subset of $\boldsymbol{or\_rank}$ and the $\boldsymbol{andCPC}$ outputted by the ANDPC algorithm in Phase III is a subset of $\boldsymbol{CPC}$. Therefore, Phase II (Line 2 of Algorithm 1) and Phase III (Lines 3 and 4 of Algorithm 1) are interchangeable, and their order does not affect the results of the DCMB algorithm.

4) Identifying Spouses in Phase IV: Phase IV (Lines 5–15 in Algorithm 1) learns the spouses of C, and the idea of the phase is the same as the spouse learning of the existing MB algorithms [4]. Based on the PC corrected by Phases II and III, for each feature $X \in \boldsymbol{PC}$, first, at Phase IV, DCMB uses IdenCPC to find the PC of X [denoted as $\boldsymbol{PC}(X)$]. Then, for each feature Y in $\boldsymbol{PC}(X)$, if there exists a subset $\boldsymbol{S}$ within $\boldsymbol{F} \setminus \{C, X, Y\}$ ($\boldsymbol{S}$ was identified and stored in the

Algorithm 4 ANDPC

```
Input: $k\_and \in [0,1]$; $\boldsymbol{CPC}$
Output: $\boldsymbol{andCPC}$: corrected CPC by the "AND" rule
1: Initialize $\boldsymbol{andCPC} = \boldsymbol{CPC}$
   /*Ascending order, $F_1$ has the lowest dependency*/
2: $\langle F_1, \ldots, F_{|\boldsymbol{CPC}|}\rangle \longleftarrow \boldsymbol{CPC}$
3: for i $= 1$ to $R(|\boldsymbol{CPC}| \times k\_and)$ do
4:   $[\boldsymbol{or\_rank2}, \boldsymbol{CPC2}] = IdenCPC(F_i)$
5:   if $C \notin \boldsymbol{CPC2}$ then
6:     $\boldsymbol{andCPC} = \boldsymbol{andCPC} \setminus \{F_i\}$
7:   end if
8: end for
```

IdenCPC algorithm) such that both $C \perp\!\!\!\perp Y|\boldsymbol{S}$ and $C \not\perp\!\!\!\perp Y|\boldsymbol{S}\cup\{X\}$ hold, Y is a spouse of C with regard to X. Finally, in Line 16 in Algorithm 1, we obtain the MB of C.

5) Tracing DCMB: We provide a tracing example to show how DCMB works in Section S-1 in the Supplementary Material.

V. Proposed SA-DCMB Algorithm

The values of parameters k_or and k_and have great influences on the performance of DCMB. In real-world applications, it is difficult to specify the suitable values of the two parameters for DCMB for different tasks. Then the question arises, whether DCMB can automatically determine a set of optimal values of k_or and k_and simultaneously? In this section, we propose the SA-DCMB algorithm to tackle this problem. SA [32] is an effective greedy search method to find the global optimum in the presence of a large number of local optima. In theory, according to the convergence analysis of Markov process, the probability of SA converging to the global solution is 1 with the extension of SA procedure [33].

The pseudo code of SA-DCMB is shown in Algorithm 5. In SA-DCMB, t is analogous to temperature in an annealing system. The value of t is high at the beginning, and then the value is gradually decreased according to an "annealing schedule." The parameter r is used to control the descent rate of t. The smaller the r, the faster t falls. The whole search process stops when t falls to t_min. In Algorithm 5, we assume that the search process of SA-DCMB is carried out in a 3-D coordinate system. The X-axis and Y-axis represent parameters k_or and k_and, respectively. Parameters h_1 and h_2 denote the unit scale sizes of the X-axis and Y-axis, respectively. In addition, the value on the Z-axis denotes the classification accuracy that is obtained by function $J(k\_\text{or}, k\_\text{and})$. The function $J()$ consists of four steps as follows.

1) First, the training dataset D_1 and validation dataset D_2 are obtained from the training dataset D (the proportion of samples in dataset D_1 and dataset D_2 is 9:1).
2) Next, the MB of the class variable C is learned using DCMB, and $\boldsymbol{MB}(C) = \text{DCMB}(C, D_1, k\_\text{or}, k\_\text{and})$.
3) Then, the classifier $f(\cdot)$ (such as naive Bayes (NB) and K-nearest neighbor (KNN) classifiers) is trained using all the features in $\boldsymbol{MB}(C)$ on dataset D_1.

Algorithm 5 SA-DCMB

```
Input: $t$: temperature; $t\_min$: the lower limit of $t$;
       $C$; $D$: training dataset; $r \in [0,1]$;
Output: MB of $C$
1: $i \leftarrow 1$, $\boldsymbol{LOS} = \emptyset$ /*locally optimal solution set*/
2: Randomly initialize $k\_or_{(i)} \in [0,1]$ and $k\_and_{(i)} \in [0,1]$
3: Implement $J((k\_or_{(i)}, k\_and_{(i)}))$ and initialize $h_1, h_2$
4: while $t > t\_min$ do
5:   $\boldsymbol{CanPS} = \{(k\_or_{(i)} \pm h_1, k\_and_{(i)} \pm h_2)\}$
6:   loop
7:     if $\boldsymbol{CanPS} = \emptyset$ then
8:       goto 30 /*stop search*/
9:     end if
10:    $Yset = \arg\max_{Xset \in \boldsymbol{CanPS}} J(Xset)$
11:    if $Yset \notin \boldsymbol{LOS}$ then
12:      goto 17
13:    else
14:      $\boldsymbol{CanPS} = \boldsymbol{CanPS} \setminus \{Yset\}$
15:    end if
16:  end loop
17:  $dE = J(Yset) - J((k\_or_{(i)}, k\_and_{(i)}))$
18:  if $dE \geqslant 0$ then
19:    $i = i+1$, $(k\_or_{(i)}, k\_and_{(i)}) \leftarrow Yset$
20:  else
21:    $\boldsymbol{LOS} = \boldsymbol{LOS} \cup \{(k\_or_{(i)}, k\_and_{(i)})\}$
22:    if $exp(dE/t) > random(0,1)$ then
23:      $t = r * t$ /*cooling annealing*/
24:      $i = i+1$, $(k\_or_{(i)}, k\_and_{(i)}) \leftarrow Yset$
25:    else
26:      goto 30 /*stop search*/
27:    end if
28:  end if
29: end while
30: Select best parameter $(k\_or_{(best)}, k\_and_{(best)})$ from $\boldsymbol{LOS}$
31: $\boldsymbol{MB} = \text{DCMB}(C, k\_or_{(best)}, k\_and_{(best)}; D)$
```

4) Finally, the class labels in D_2 are predicted using $f(\cdot)$ and the classification accuracy is obtained.

The details of SA-DCMB are described as follows.

First, we randomly initialize parameter k_or and k_and in the range of [0,1] (Line 2 in Algorithm 5) and call the function $J(\cdot)$ (Line 3).

Based on the obtained ***or_rank*** and ***CPC***, h_1 and h_2 are initialized to $1/|\boldsymbol{or\_rank}|$ and $1/|\boldsymbol{CPC}|$, respectively. h_1 and h_2 ensure that features in the sets of ***or_rank*** and ***CPC*** are traversed one-by-one throughout the process of implementing *ORPC* (see Algorithm 3) and *ANDPC* (see Algorithm 4).

Second, SA-DCMB implements the SA search process for searching for the best parameter $(k\_\text{or}_{(\text{best})}, k\_\text{and}_{(\text{best})})$ in the range of [0,1] (Lines 4–29). SA-DCMB initializes the candidate parameter set (***CanPS***) (Line 5), including four parameter combinations: $[(k\_\text{or}_{(i)} + h_1, k\_\text{and}_{(i)})$, $(k\_\text{or}_{(i)}, k\_\text{and}_{(i)} + h_2)$, $(k\_\text{or}_{(i)} - h_1, k\_\text{and}_{(i)})$, and $(k\_\text{or}_{(i)}, k\_\text{and}_{(i)} - h_2)]$. Then, SA-DCMB traverses ***CanPS*** (Lines 6–16) to select the parameter set (k_or, k_and) within $\boldsymbol{CanPS}\setminus\boldsymbol{LOS}$ for obtaining maximum classification accuracy.

TABLE I
SUMMARY OF BENCHMARK BNS FOR VALIDATING DCMB

Network	Num. Vars	Num. Edges	Max In/out-Degree	Min/Max |PCset|	Variable Domain
Child3	60	79	3/7	1/8	2-6
Insurance10	270	556	5/8	1/11	2-5
Alarm	37	46	4/5	1/6	2-4
Alarm3	111	149	4/5	1/6	2-4

Next, if $J(Yset) \geq J((k\_or_{(i)}, k\_and_{(i)}))$, SA-DCMB will choose parameter *Yset* as a new search direction (Line 19); otherwise, the current set of $(k\_or_{(i)}, k\_and_{(i)})$ is considered as the best parameter set. Since the probability of cooling with an energy difference of *dE* is *exp*(*dE*/*t*), SA-DCMB can accept the poor parameter (k_or, k_and) and starts annealing once if and only if *exp*(*dE*/*t*) > *random* (0,1) holds (Lines 22–24), and *random* (0,1) is used to generate a random number from 0 to 1.

Finally, SA-DCMB selects the best (k_or, k_and) within ***LOS*** (Line 30), and the MB of C is obtained (Line 31).

VI. EXPERIMENTS

In this section, we first validate that DCMB achieves significant improvements on MB learning accuracy on benchmark BN datasets, and then we analyze the parameters k_or and k_and. Finally, we evaluate the effectiveness of SA-DCMB on real-world datasets.

All the experiments are conducted on a computer with Intel Core i5-8400 2.80-GHz CPU and 16-GB memory.

A. Benchmark BN Datasets for Validating DCMB

1) Datasets: We use four benchmark BNs with different numbers of variables in our experiments, and the details of the four benchmark BNs are summarized in Table I.[1] For each benchmark BN network, we randomly generate three datasets, including 500 data instances, 1000 data instances, and 5000 data instances, respectively.

2) Comparison Methods: We compare DCMB with four state-of-the-art causal feature selection algorithms, including FBED^K [23], PCMB [9], BAMB [29], and CCMB [11].[2] Note that PCMB adopts the "AND" rule, while CCMB uses the "OR" rule.

3) Evaluation Metrics: For the benchmark BN networks, the MB of each feature can be read from those networks. Accordingly, in the experiments, we evaluate the algorithms using the following metrics.

1) $Precision$: The precision metric denotes the number of true positives in the output (i.e., the features in the output of an algorithm belonging to the true MB of a given target in a test DAG) divided by the number of features in the output of the algorithm.
2) $Recall$: The recall metric represents the number of true positives in the output divided by the number of true positives (the number of the true MB of a given target) in a test DAG.
3) $F1$: $F1 = 2{*}Precision{*}Recall/(Precision{+}Recall)$. The $F1$ score is the harmonic average of the precision and recall, where $F1 = 1$ is the best case (perfect precision and recall) while $F1 = 0$ is the worst case.
4) $Time$: We report running time (in seconds) as the efficiency measure of different algorithms.

[1]Those benchmark BN networks are publicly available at http://www.bnlearn.com/bnrepository/

[2]The source codes are available at https://github.com/kuiy/CausalFS

TABLE II
EXPERIMENTAL RESULTS ON BENCHMARK DATASETS FOR VALIDATING THE EFFICIENCY OF THE DUAL CORRECTION STRATEGY (TIME METRIC, IN SECONDS)

Network	Sample	PCMB	FBED	BAMB	CCMB	DCMB
Alarm	500	0.004	**0.000**	0.001	0.012	0.006
	1000	0.009	**0.001**	0.002	0.027	0.013
	5000	0.062	**0.005**	0.016	0.149	0.074
Alarm3	500	0.007	**0.001**	0.002	0.067	0.013
	1000	0.015	**0.002**	0.005	0.143	0.028
	5000	0.104	**0.012**	0.028	0.784	0.191
Child3	500	0.006	**0.001**	0.002	0.025	0.007
	1000	0.015	**0.001**	0.003	0.055	0.018
	5000	0.114	**0.007**	0.023	0.340	0.135
Insurance10	500	0.025	**0.002**	0.008	0.391	0.033
	1000	0.064	**0.005**	0.018	0.854	0.078
	5000	0.550	**0.031**	0.123	5.376	0.854

4) Implementation Details:

1) All the algorithms are implemented in C/C++. For the FBED^K algorithm, the value of K is set to 1, which is enough to make FBED^K converge.
2) The CI tests are G^2 tests with a statistical significance level of 0.01.
3) For an algorithm, we identify the MBs of all the features in each BN and report the average results of $F1$, precision, recall, and time.

a) Experimental results of the dual correction strategy: In this section, we validate the effectiveness and the efficiency of the dual correction strategy of the DCMB algorithm.

We compare DCMB with the four state-of-the-art MB learning algorithms, BAMB, CCMB, FBED, and PCMB on the benchmark datasets with 500, 1000, and 5000 samples, respectively. The experimental results are shown in Figs. 2–4 and Table II. Specifically, we traverse k_or from 0 to 1 and k_and from 0 to 1 simultaneously, and we record the change process of precision, recall, and $F1$ metrics as shown in Figs. 2–4, respectively. For Figs. 2–4, [(a)–(d)], [(e)–(h)], and [(i)–(l)] denote the experimental results with 500, 1000, and 5000 samples, respectively. In addition, in Table II, we also present the time metric, and note that we record the time when the $F1$ metric of DCMB reaches the maximum as the running time of DCMB on a dataset.

From Figs. 2–4 and Table II, our observations are summarized as follows.

1) The precision metric of PCMB is high on most datasets, since PCMB uses the AND rule to correct the false positive errors, and even if some true positives are mistakenly deleted, the precision metric may not go down. Similarly, the recall metric of CCMB is high on most datasets, since CCMB adopts the OR rule to correct the false negative errors, and even if some true negatives are wrongly added to MB of the class variable, the recall metric cannot decrease. However, the $F1$

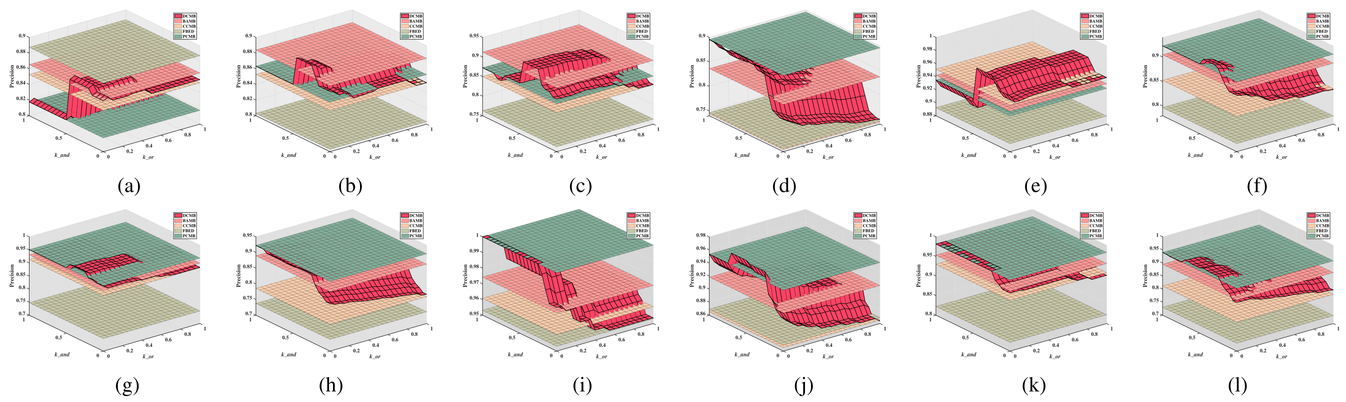

Fig. 2. Experimental results on benchmark datasets for validating the effectiveness of the dual correction strategy (precision metric). (a), (e), and (i) Alarm. (b), (f), and (j) Alarm3. (c), (g), and (k) Child3. (d), (h), and (l) Insurance10.

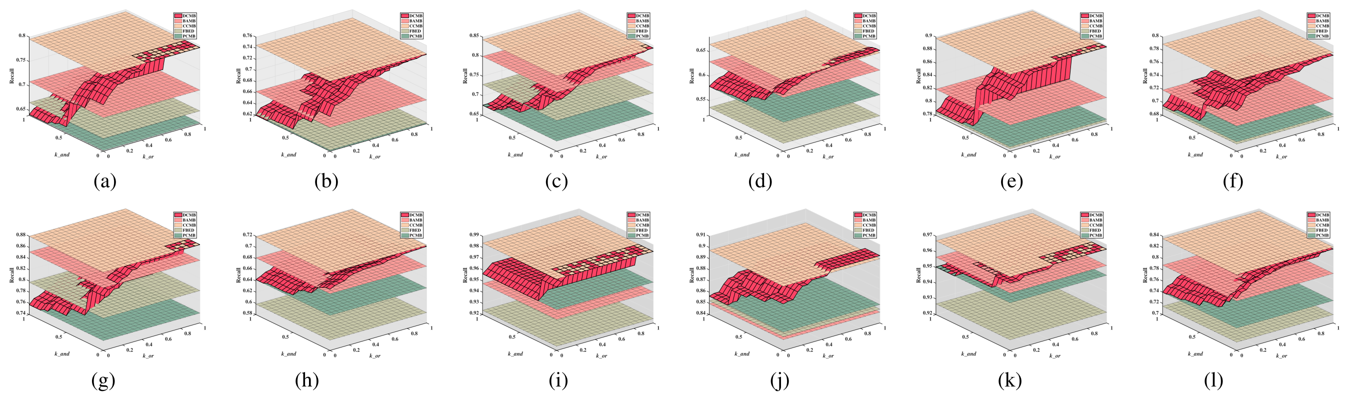

Fig. 3. Experimental results on benchmark datasets for validating the effectiveness of the dual correction strategy (recall metric). (a), (e), and (i) Alarm. (b), (f), and (j) Alarm3. (c), (g), and (k) Child3. (d), (h), and (l) Insurance10.

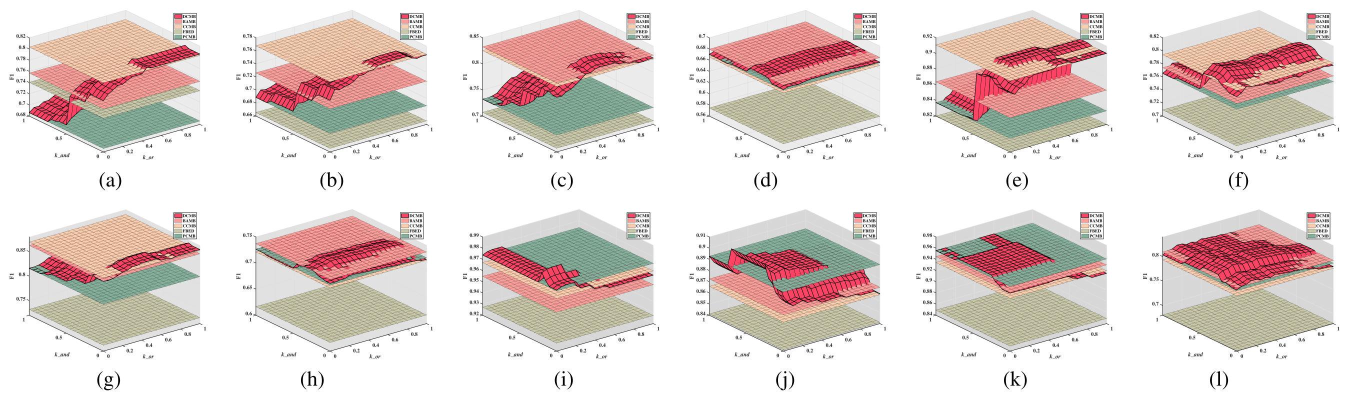

Fig. 4. Experimental results on benchmark datasets for validating the effectiveness of the dual correction strategy ($F1$ metric). (a), (e), and (i) Alarm. (b), (f), and (j) Alarm3. (c), (g), and (k) Child3. (d), (h), and (l) Insurance10.

metrics of PCMB and CCMB are not very high on most datasets, while DCMB always gets the highest $F1$ value on all the datasets, since PCMB may delete the true MB features and CCMB may select false MB features, which reduce the recall of PCMB and precision of CCMB, respectively.

2) With the increase in the value of k_and of DCMB, the precision metric of DCMB first rises and then declines; the recall metric of DCMB is positively correlated with k_or. This is because removing true MB features from MB(C) (the MB of the class variable) may reduce the precision metric, while adding false MB features to MB(C) cannot reduce the recall metric. Particularly, when the sample size is small (such as 500 and 1000 samples), many CI tests become unreliable during the MB identification process. In this case, if the value of

k_and of DCMB is large, some true positives might be deleted from MB(C), i.e., the precision metric declines, as shown in Fig. 2(a)–(g).

3) When the sample size is small (such as 500 and 1000 samples), the value of k_or of DCMB is large and the value of k_and of DCMB is small, the $F1$ metric of DCMB can reach its peak. In contrast, when the sample size is large (such as 5000 samples), the value of k_or is small and the value of k_and is large, the $F1$ metric of DCMB can obtain the highest value (we analyze this phenomenon in detail in Section VI-B).

4) On all the datasets, the time cost of DCMB is always much lower than that of CCMB and slightly higher than that of PCMB, since their time cost mainly depends on the number of calls made to the PC learning algorithm (e.g. IdenCPC, Algorithm 2). Specifically, for the selective removal/retrieval in the dual correction strategy, the ORPC and ANDPC algorithms need to implement the PC learning algorithm $R(|\boldsymbol{or\_rank}| \times k\_\text{or})$ times (Lines 3 and 4 in Algorithm 3) and $R(|\boldsymbol{CPC}| \times k\_\text{and})$ times (Lines 3 and 4 in Algorithm 4), respectively. Thus, DCMB needs to run the PC learning algorithm $[R(|\boldsymbol{CPC}| \times k\_\text{and}) + R(|\boldsymbol{or\_rank}| \times k\_\text{or})]$ times. In addition, PCMB and CCMB call the PC learning algorithm $|\boldsymbol{CPC}|$ times [9] and $(|\boldsymbol{F}| - |\boldsymbol{CPC}|)$ times [11], respectively. Generally, $|\boldsymbol{CPC}| < (R(|\boldsymbol{CPC}| \times k\_\text{and}) + R(|\boldsymbol{or\_rank}| \times k\_\text{or})) \ll (|\boldsymbol{F}| - |\boldsymbol{CPC}|)$ holds, i.e., the number of times that DCMB needs to call the PC learning algorithm is slightly higher than PCMB but significantly less than CCMB.

b) Detailed experimental results: We present more experimental results in Section S-2 of the Supplementary Material, where DCMB is compared with more baselines on more benchmark datasets.

c) The rationale of the dual correction strategy of DCMB: In this section, we use the experimental results on the benchmark datasets to demonstrate the rationale of the dual correction strategy of DCMB.

Using the IdenCPC algorithm (see Algorithm 2), we conducted experiments on the Alarm BN with 37 variables including 1000 samples. Specifically, we take each variable as the class variable C in turn to identify their PC by IdenCPC. Then, we record the relevancy value between other variables and C after calling IdenCPC. As shown in Fig. 5, we select ten variables (denoted as ordinate) for demonstration and the PC learning of those variables encountered mistakes. Since the PC of each variable can be read off from the benchmark BN, false positives and false negatives can be distinguished from true positives and true negatives, respectively. The pink circle, light blue circle, red cross, and blue cross denote true positives, true negatives, false positives, and false negatives, respectively.

For a CI test of F_i and C conditioning on $\boldsymbol{S}$, when the relevancy value calculated by the function dep() (see Algorithm 2) is greater than 0, $F_i \not\perp\!\!\!\perp C|\boldsymbol{S}$ holds; otherwise, $F_i \perp\!\!\!\perp C|\boldsymbol{S}$. We can see that the relevancy values of false positives are closer to 0^+ than true positives. Similarly, the relevancy values of false negatives are closer to 0^- than true negatives.

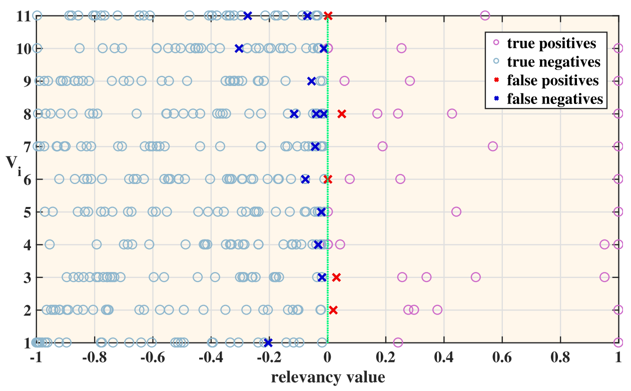

Fig. 5. Example of relevancy value distribution when conducting experiments on the Alarm benchmark BN dataset.

Thus, the ORPC algorithm (see Algorithm 3) selects the variable with a greater relevancy value with C for correction, and the ANDPC algorithm (see Algorithm 4) selects the variable with a smaller relevancy value with C for correction.

d) Experimental results of the single correction strategy: In Section S-3 in the Supplementary Material, we validate the effectiveness of the single correction strategy using either the AND rule or the OR rule, respectively.

B. Conjectures of the Value of k_or and k_and

In Phases II and III of the DCMB algorithm, it is critical to specify suitable values of k_or and k_and for the ORPC and ANDPC algorithms.

For the ORPC and ANDPC algorithms, we sort the features in $\boldsymbol{or\_rank}$ in descending order, while arranging the ones in $\boldsymbol{CPC}$ in ascending order. Then these two algorithms selectively choose features from $\boldsymbol{or\_rank}$ and $\boldsymbol{CPC}$, respectively, using the parameters k_or and k_and. From the experimental results in Section VI-A above, for the two parameters, we have the following conjectures.

1) Conjecture 1: If the size of data samples is much smaller than the dimensionality, the value of k_or bigger than 0.5 is better, while the value of k_and smaller than 0.5 is better.

2) Conjecture 2: If the size of data samples is much bigger than the dimensionality, the value of k_or smaller than 0.5 is better, while the value of k_and bigger than 0.5 is better.

The rationale of Conjecture 1 is that when a dataset with small-sized data samples and high dimensionality, many CI tests may be unreliable, resulting in many true PC of C being discarded. In this case, recovering discarded true positives is more important than removing false positives. A large k_and value will make more true PC be discarded using the AND rule. Instead, we need to use a large k_or value to recover more discarded PC.

For Conjecture 2, when a dataset has a large number of data samples, most CI tests are reliable. In this case, many false positives may be selected while few true PC of C are discarded. Thus, at this time removing false positives is more critical than recovering discarded true positives. Under this situation, a large k_and value and a small k_or will be suitable.

In the experiments using real-world datasets (i.e., Section VI-C), these two conjectures are particularly beneficial

TABLE III
DESCRIPTION OF REAL-WORLD DATASETS USED IN THE EXPERIMENTS

No.	Dataset	Number of instances	Number of classes	Number of features
1	leuk	72	2	7,070
2	arcene	100	2	10,000
3	prostate	102	2	6,033
4	lungcancer	181	2	12,533
5	sonar	208	2	60
6	ovarian	216	2	2,190
7	spect	267	2	22
8	breastcancer	286	2	17,816
9	dexter	300	2	20,000
10	semeion	1,593	10	256
11	madelon	2,000	2	500
12	spambase	4,601	2	57

to the initialization of parameters k_or and k_and of the SA-DCMB algorithm.

C. Real-World Datasets for Validating SA-DCMB

1) Datasets: We use 12 real-world datasets with a variety of dimensions and sample sizes in our experiments, and these datasets are from the UCI Machine Learning Repository, KDD2008 Gene Expression Data and NIPS2003 feature selection challenge datasets. Details of the datasets are shown in Table III, and we can see that *leuk*, *arcene*, *prostate*, *lungcancer*, *ovarian*, *breastcancer*, and *dexter* belong to high-dimensional small sample datasets, and *semeion* belongs to multi-class dataset.

2) Comparison Methods: We compare SA-DCMB with 16 other algorithms, including 12 causal feature selection algorithms, IAMB [5], FBED^K [23], MMMB [12], PCMB [9], HITON-MB [24], MBOR [26], IPCMB [10], STMB [27], BAMB [29], CCMB [11], EEMB [30][3], and SRMB [28], and four well-established feature selection algorithms, least absolute shrinkage and selection operator (LASSO) [34], fast correlation-based filter (FCBF) [35], quadratic programming feature selection (QPFS) [36], and fuzzy similarity and entropy (FSAE) [37].

3) Evaluation Metrics: We use the following metrics for the feature selection evaluation.

1) *Accuracy*: We report the classification accuracy of the NB classifier and the KNN classifier for SA-DCMB and all the compared algorithms. Classification accuracy is the percentage of the correctly classified test instances in all test instances.
2) *Compactness*: Compactness is the size of the feature subset selected by an algorithm.
3) *Time*: We measure the efficiency of an algorithm using the total runtime (in seconds) of this algorithm and classifier.

4) Implementation Details:

1) The running details of all causal feature selection methods are consistent with those on benchmark BN datasets.
2) LASSO, FCBF, QPFS, and FSAE are implemented in MATLAB. The information threshold of FCBF is set to 0.01. As for LASSO, QPFS, and FSAE, we choose the top N features where N is the size of the MB obtained by SA-DCMB on each dataset.
3) We apply tenfold cross-validation for all the datasets and use two classifiers, i.e., NB and KNN to compute their classification accuracies achieved using the selected feature subsets. The value of k for the KNN classifier is set to 10 and KNN uses the linear kernel.
4) Parameters r and t_min of SA-DCMB are set to 0.8 and 20, respectively. Based on two conjectures in Section VI-B, given a dataset, we can get a rough 2-D empirical interval $\Omega = \{(x, y)|x \in [a, b], y \in [c, d], 0 \leqslant a \leqslant b \leqslant 1, 0 \leqslant c \leqslant d \leqslant 1\}$. Obviously, the range of Ω is diverse on different types of datasets. Considering the randomness of SA, we randomly initialize three pairs of parameters $(k\_\text{or}, k\_\text{and}) \in \Omega$ to implement SA-DCMB three times, and then record the highest classification accuracy as the final result on a dataset.

[3]The source codes of these algorithms are available at https://github.com/kuiy/CausalFS

In Tables IV–IX, "-" denotes that an algorithm fails to generate any output with the corresponding dataset after running more than one day and "*" denotes that no feature is selected by an algorithm. In addition, the best results are highlighted in bold face.

a) Comparison of SA-DCMB to causal feature selection algorithms: In this section, we report the results obtained by SA-DCMB and the other 12 state-of-the-art MB learning algorithms, including IAMB, FBED^K, MMMB, PCMB, HITON-MB, MBOR, IPCMB, STMB, BAMB, CCMB, EEMB, and SRMB.

1) *Accuracy Metric:* Table IV summarizes the classification accuracy of SA-DCMB against IAMB, FBED^K, MMMB, PCMB, HITON-MB, MBOR, IPCMB, STMB, BAMB, CCMB, EEMB, and SRMB using NB and KNN classifiers. We can see that SA-DCMB is superior to the other algorithms on most datasets using both NB and KNN. In particular, using NB, the accuracy of the SA-DCMB algorithm has reached 100% on the *leuk* dataset; on the *arcene* and *semeion* datasets, the classification accuracy of SA-DCMB is more than 10% higher than that of the other algorithms. Furthermore, using KNN, the classification accuracy of SA-DCMB is approximately 10% higher than that of the other algorithms on the *arcene* dataset. On the whole, the algorithms using the OR rule have high classification accuracy on high-dimensional and small-sized datasets (such as *leuk*, *arcene*, *prostate*, *lungcancer*, *ovarian*, *breastcancer*, and *dexter*), since many true MB features are retrieved; when the sample size of the dataset is sufficient (such as *sonar*, *spambase*, and *semeion*), the algorithms using the AND rule have better classification accuracy due to some false MB features being removed. Since SA-DCMB uses the dual correction strategy, it can delete the false MB features and recover the true MB features simultaneously, and this strategy can also avoid removing the true MB features or recalling the false MB features. Thus, SA-DCMB performs well on all the datasets. Note that the *semeion* dataset is dense and multi-class, so a large number of true MB features

TABLE IV
ACCURACY (IN %) OF SA-DCMB AND OTHER CAUSAL FEATURE SELECTION ALGORITHMS

Classifier	Dataset	IAMB	FBED	MMMB	PCMB	HITON-MB	MBOR	IPCMB	STMB	BAMB	CCMB	EEMB	SRMB	SA-DCMB
NB	leuk	91.61	91.61	97.32	95.89	97.32	97.32	95.89	37.50	97.32	97.14	97.32	97.14	**100.00**
	arcene	75.16	77.16	74.36	66.21	73.36	69.14	64.21	67.05	72.27	75.45	74.36	77.16	**86.27**
	prostate	91.00	91.00	89.09	89.09	92.00	**95.00**	68.91	61.09	92.00	94.00	92.00	94.00	94.00
	lungcancer	96.17	96.17	96.17	91.17	95.61	**98.89**	*	*	94.47	95.56	95.61	96.17	98.36
	ovarian	87.06	87.51	89.81	81.52	90.28	90.28	81.08	45.78	90.37	91.17	88.90	90.28	**94.03**
	breastcancer	80.05	80.38	84.37	77.64	83.21	84.61	74.47	26.91	80.71	-	84.27	-	**85.65**
	dexter	79.33	79.33	86.00	85.00	85.67	88.67	80.67	86.33	86.33	88.67	85.33	88.67	**89.67**
	madelon	59.80	60.25	58.85	56.05	58.95	59.30	58.80	59.55	58.70	59.95	60.15	60.25	**61.30**
	sonar	71.55	72.48	69.20	**77.55**	68.74	76.55	**77.55**	74.46	70.81	69.20	69.69	71.55	**77.55**
	spambase	90.09	90.09	88.09	88.13	88.22	88.89	87.96	88.39	88.89	88.46	**90.31**	88.46	88.57
	spect	79.03	79.03	77.52	66.97	75.68	78.67	43.56	78.65	74.58	79.04	75.68	79.04	**79.82**
	semeion	54.18	54.18	45.08	68.01	50.19	58.90	70.39	44.79	-	-	-	-	**81.93**
KNN	leuk	91.61	93.04	93.21	93.21	93.21	94.46	94.46	82.14	91.61	94.29	94.46	94.29	**95.71**
	arcene	68.05	69.05	70.14	68.12	77.36	66.05	66.12	72.16	68.05	72.94	72.96	72.16	**86.27**
	prostate	91.00	91.00	87.09	89.09	93.00	93.00	49.36	78.27	93.00	93.00	93.00	93.00	**95.00**
	lungcancer	96.70	96.70	96.70	88.92	95.61	97.81	*	*	94.47	97.81	95.61	96.70	**98.36**
	ovarian	86.62	88.01	88.42	81.06	89.85	90.30	81.08	78.18	89.87	90.30	89.85	90.30	**92.16**
	breastcancer	79.68	78.97	84.68	73.42	83.57	85.28	70.92	83.18	81.42	-	83.93	-	**86.75**
	dexter	76.00	76.00	84.00	80.00	81.67	86.33	81.00	85.67	83.00	85.67	84.00	85.67	**87.33**
	madelon	**64.80**	63.25	58.55	52.00	59.35	59.05	58.40	60.40	63.55	60.50	63.10	63.25	59.85
	sonar	73.01	66.81	78.85	74.01	75.94	74.10	74.01	**79.27**	74.99	78.85	77.37	78.85	77.80
	spambase	89.30	89.30	92.39	92.50	92.13	92.24	92.15	92.26	92.50	92.24	92.26	92.50	**92.78**
	spect	80.14	80.14	79.80	63.20	77.97	76.80	31.49	76.30	77.61	79.80	77.97	79.80	**80.19**
	semeion	51.36	51.36	89.71	78.48	89.71	57.28	78.48	89.71	-	-	-	-	**90.84**

TABLE V
COMPACTNESS OF SA-DCMB AND OTHER CAUSAL FEATURE SELECTION ALGORITHMS

Classifier	Dataset	IAMB	FBED	MMMB	PCMB	HITON-MB	MBOR	IPCMB	STMB	BAMB	CCMB	EEMB	SRMB	SA-DCMB
NB	leuk	2.00	2.00	5.40	4.30	5.10	6.40	3.90	339.60	9.50	30.30	9.00	30.10	10.30
	arcene	4.00	4.00	4.00	1.40	4.00	3.50	1.70	28.40	4.40	5.60	4.20	5.80	4.60
	prostate	3.00	3.50	4.10	1.20	3.20	7.40	0.60	461.10	3.00	11.40	3.00	10.00	8.60
	lungcancer	3.20	3.20	3.40	2.10	3.30	119.10	*	*	4.10	187.20	3.10	165.10	16.40
	ovarian	5.00	5.00	5.80	2.40	6.00	6.00	2.10	556.10	7.40	13.70	5.70	14.20	12.10
	breastcancer	5.00	5.00	9.20	1.90	8.40	19.20	1.40	1733.00	8.80	-	8.30	-	15.70
	dexter	5.00	5.00	9.50	6.90	9.30	21.70	5.80	57.50	10.20	26.60	9.00	28.30	21.50
	madelon	7.00	7.20	5.70	1.50	5.10	6.80	3.60	24.00	7.40	11.50	6.80	13.20	9.10
	sonar	2.00	2.00	15.30	5.80	15.20	2.70	5.80	14.00	15.60	15.30	13.70	14.70	5.90
	spambase	9.00	9.00	43.90	41.30	43.10	37.40	42.00	50.00	35.40	49.00	31.60	46.00	48.10
	spect	3.00	3.00	3.00	0.80	2.70	4.60	0.40	1.80	2.90	3.70	2.70	3.70	2.60
	semeion	5.00	5.00	206.50	61.30	201.50	8.10	55.00	212.40	-	-	-	-	155.10
KNN	leuk	2.00	2.00	5.40	4.30	5.10	6.40	3.90	339.60	9.50	30.30	9.00	30.10	10.20
	arcene	4.00	4.00	4.00	1.40	4.00	3.50	1.70	28.40	4.40	5.60	4.20	5.80	4.80
	prostate	3.00	3.50	4.10	1.20	3.20	7.40	0.60	461.10	3.00	11.40	3.00	10.00	6.50
	lungcancer	3.20	3.20	3.40	2.10	3.30	119.10	*	*	4.10	187.20	3.10	165.10	21.10
	ovarian	5.00	5.00	5.80	2.40	6.00	6.60	2.10	556.10	7.40	13.70	5.70	14.20	11.90
	breastcancer	5.00	5.00	9.20	1.90	8.40	19.20	1.40	1733.00	8.80	-	8.30	-	18.10
	dexter	5.00	5.00	9.50	6.90	9.30	21.70	5.80	57.50	10.20	26.60	9.00	28.30	15.00
	madelon	7.00	7.20	5.70	1.50	5.10	6.80	3.60	24.00	7.40	11.50	6.80	13.20	5.00
	sonar	2.00	2.00	15.30	5.80	15.20	2.70	5.80	14.00	15.60	15.30	13.70	14.70	8.50
	spambase	9.00	9.00	43.90	41.30	43.10	37.40	42.00	50.00	35.40	49.00	31.60	46.00	43.40
	spect	3.00	3.00	3.00	0.80	2.70	4.60	0.40	1.80	2.90	3.70	2.70	3.70	2.70
	semeion	5.00	5.00	206.50	61.30	201.50	8.10	55.00	212.40	-	-	-	-	184.40

will be discarded [20] on this dataset, which renders that all the algorithms except SA-DCMB have poor performance. As for the latest two baselines, EEMB [30] and SRMB [28], although they have achieved excellent performance, they are still inferior to our method on the whole. EEMB optimizes the search strategy of BAMB, so that the classification accuracy of EEMB is slightly better than BAMB in general, and SRMB achieves a comparable performance against CCMB, since the feature subsets selected by them are similar across all the datasets.

2) *Compactness Metric:* From Table V, we can observe that IPCMB and STMB do not select any features on the *lungcancer* dataset. This is because IPCMB and STMB use the same PC learning algorithm, RecognizePC [38], for finding the PC set of C, and RecognizePC adopts the backward strategy to remove features that are independent of C from the candidate PC set of C. When the CI tests are unreliable, the backward strategy may result in an empty candidate PC set, whereas the forward strategy aims to constantly add features that depend on C to the candidate PC set of C, which can avoid the situation of an empty candidate PC set for C. As the *lungcancer* dataset has high dimensionality and contains a small number of samples, CI tests conducted with the dataset can be unreliable [20]. This leads to the results that all true PC features have been discarded, and hence the PC set of C becomes empty. For the same reason, all spouses of C have been discarded. Nevertheless, on many datasets, the STMB algorithm can select more features than other algorithms since it identifies the spouses of C from $\boldsymbol{F} \setminus \boldsymbol{PC}(C)$ instead of the PC set of each feature in $\boldsymbol{PC}(C)$, and this adds many false spouses to $\boldsymbol{MB}(C)$. Specifically, on the *leuk*, *arcene*, *prostate*, *ovarian*, *breastcancer*, and *semeion* datasets, and STMB selects more features and achieves lower accuracy than the other methods. However, SA-DCMB always keeps selecting fewer features to achieve higher classification accuracy on all the datasets. Specifically, on most datasets, SA-DCMB selects fewer features than the algorithms using the OR rule and more features than the algorithms using the AND rule or without any rules. This is because an algorithm using the OR rule may add the false MB features to the MB of C on most datasets. On all the datasets, IAMB and FBED obtain similar feature subsets and the size of their subsets is small since

TABLE VI
TIME (IN SECONDS) OF SA-DCMB AND OTHER CAUSAL FEATURE SELECTION ALGORITHMS

Classifier	Dataset	IAMB	FBED	MMMB	PCMB	HITON-MB	MBOR	IPCMB	STMB	BAMB	CCMB	EEMB	SRMB	SA-DCMB
NB	leuk	0.09	**0.08**	0.44	2.65	0.44	0.16	0.45	0.32	0.22	624.12	0.44	631.06	16.03
	arcene	0.21	**0.14**	0.73	1.41	0.78	0.21	0.40	0.65	0.65	1872.60	0.87	1991.45	15.75
	prostate	0.12	**0.11**	0.39	0.73	0.35	0.37	0.71	0.50	0.25	626.81	0.37	646.15	15.77
	lungcancer	0.48	**0.42**	1.45	4.11	1.52	68.05	*	*	23.61	3754.12	1.69	3856.26	141.43
	ovarian	0.10	**0.08**	0.17	0.36	0.19	0.59	0.36	0.26	0.18	63.35	0.16	64.26	17.57
	breastcancer	1.35	**0.94**	5.43	11.69	6.37	423.27	61.16	54.85	14.40	-	5.47	-	985.69
	dexter	1.17	**0.74**	2.71	8.60	2.73	5.57	3.85	3.74	2.89	1921.31	4.15	1986.25	13.74
	madelon	0.23	**0.17**	0.20	0.26	0.19	0.32	0.20	0.20	0.26	4.85	0.20	4.91	0.85
	sonar	**0.00**	**0.00**	**0.00**	0.02	0.01	**0.00**	**0.00**	**0.00**	**0.00**	0.03	**0.00**	0.03	0.01
	spambase	0.06	**0.05**	4.94	98.27	10.86	11.79	28.19	28.88	62.76	12.04	26.10	13.69	9.51
	spect	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
	semeion	**0.08**	**0.08**	523.32	16702.60	1178.62	24.31	242.29	1358.71	-	-	-	-	515.32
KNN	leuk	0.09	**0.08**	0.59	3.70	0.59	0.16	0.45	0.32	0.22	624.16	0.42	630.98	25.45
	arcene	0.21	**0.14**	0.99	1.95	1.06	0.21	0.39	0.68	0.63	1868.68	0.90	1989.56	16.26
	prostate	0.13	**0.11**	0.50	0.98	0.46	0.38	0.70	0.48	0.25	627.36	0.43	643.87	21.96
	lungcancer	0.47	**0.43**	1.94	5.71	1.97	63.37	*	*	24.47	3751.81	1.88	3849.59	216.87
	ovarian	0.10	**0.08**	0.20	0.43	0.22	0.59	0.36	0.25	0.18	64.15	0.16	64.78	14.99
	breastcancer	1.42	**0.88**	7.35	16.04	8.25	430.96	65.12	41.91	13.10	-	5.55	-	926.89
	dexter	1.16	**0.79**	3.28	11.16	3.27	5.60	3.62	3.39	2.77	1909.46	4.17	1992.65	28.58
	madelon	0.25	**0.19**	0.21	0.29	0.21	0.32	0.21	0.21	0.27	4.92	0.22	4.98	0.33
	sonar	**0.00**	**0.00**	**0.00**	0.02	0.01	**0.00**	**0.00**	**0.00**	**0.00**	0.03	**0.00**	0.03	0.01
	spambase	0.15	**0.14**	5.66	109.15	11.96	12.09	28.63	29.37	63.36	12.50	26.46	14.11	11.99
	spect	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**	**0.00**
	semeion	**0.08**	**0.08**	512.79	17057.90	1162.13	24.43	242.88	1359.96	-	-	-	-	528.67

they suffer from the data inefficiency problem (i.e., many key features are independent of C conditioning on the MB feature subset currently selected). In addition, the subset of features selected by SRMB is also similar to that selected by CCMB.

3) *Time Metric:* In Table VI, we find that the methods using either rule, especially the OR rule, are computationally expensive. For example, on most datasets, PCMB, MBOR, IPCMB, CCMB, and SRMB are slower than the other methods without using either rule. Since SA-DCMB selectively chooses features from ***or_rank*** and ***CPC*** to correct the false positive errors and the false negative errors, it may be more efficient than other algorithms using two rules. Specifically, on the *breastcancer* and *semeion* datasets, SA-DCMB only costs hundreds of seconds, while CCMB and SRMB spend more than one day; on the *spambase* and *semeion* datasets, SA-DCMB is significantly faster than algorithms using two rules. Since EAMB improves the search strategy of BAMB, EEMB is significantly faster than BAMB on some high-dimensional datasets (such as *lungcancer* and *breastcancer*).

b) Comparison of SA-DCMB with non-causal feature selection algorithms: In this section, we report and discuss the experimental results of the SA-DCMB algorithm compared with the four well-established feature selection methods, LASSO, FCBF, QPFS, and FSAE.

1) *Accuracy Metric:* From Table VII, it can be seen that the SA-DCMB algorithm has higher accuracy than the other three algorithms using both NB and KNN classifiers on most datasets. Specifically, SA-DCMB is never worse than LASSO in classification accuracy using both NB and KNN on each dataset. On the datasets with a large number of features and a small number of samples (such as *arcene*, *leuk*, *prostate*, *lungcancer*, *ovarian*, *breastcancer*, and *dexter*), the advantage of SA-DCMB in classification accuracy is more obvious. In particular, on the *arcene* dataset, the classification accuracy of SA-DCMB is more than 20% higher than that of the other algorithms using KNN.

TABLE VII
ACCURACY (IN %) OF SA-DCMB AND NON-CAUSAL FEATURE SELECTION ALGORITHMS

Classifier	Dataset	FCBF	LASSO	QPFS	FSAE	SA-DCMB
NB	leuk	97.14	87.68	95.71	85.89	**100.00**
	arcene	70.34	71.05	55.12	56.01	**86.27**
	prostate	**94.00**	93.00	60.64	93.00	**94.00**
	lungcancer	77.25	93.36	92.81	97.25	**98.36**
	ovarian	90.74	83.38	88.01	80.06	**94.03**
	breastcancer	77.27	83.92	-	77.97	**85.65**
	dexter	88.33	82.67	82.33	50.00	**89.67**
	madelon	57.00	59.85	60.35	58.90	**61.30**
	sonar	73.13	69.06	71.67	60.03	**77.55**
	spambase	90.00	85.94	89.39	**90.41**	88.57
	spect	77.53	73.43	79.05	79.42	**79.82**
	semeion	80.35	83.24	**84.87**	84.12	81.93
KNN	leuk	**95.71**	86.25	92.86	78.93	**95.71**
	arcene	67.32	64.32	52.19	54.01	**86.27**
	prostate	94.09	91.09	53.09	94.00	**95.00**
	lungcancer	93.36	87.31	88.95	97.78	**98.36**
	ovarian	90.80	87.58	88.46	81.49	**92.16**
	breastcancer	**88.13**	81.15	-	73.09	86.75
	dexter	80.00	77.67	82.33	50.00	**87.33**
	madelon	55.70	56.55	57.95	55.85	**59.85**
	sonar	73.51	67.22	75.51	65.20	**77.80**
	spambase	90.31	90.31	92.09	92.04	**92.78**
	spect	78.64	79.44	74.50	78.27	**80.19**
	semeion	82.56	89.64	90.40	90.53	**90.84**

TABLE VIII
COMPACTNESS OF SA-DCMB AND NON-CAUSAL FEATURE SELECTION ALGORITHMS

Classifier	Dataset	FCBF	LASSO	QPFS	FSAE	SA-DCMB
NB	leuk	52.30	10.00	10.00	10.00	10.30
	arcene	33.50	5.00	5.00	5.00	4.60
	prostate	41.90	9.00	9.00	9.00	8.60
	lungcancer	275.60	16.00	16.00	16.00	16.40
	ovarian	12.70	12.00	12.00	12.00	12.10
	breastcancer	205.70	16.00	-	16.00	15.70
	dexter	41.30	22.00	22.00	22.00	21.50
	madelon	2.00	9.00	9.00	9.00	9.10
	sonar	2.30	6.00	6.00	6.00	5.90
	spambase	10.60	48.00	48.00	48.00	48.10
	spect	4.30	3.00	3.00	3.00	2.60
	semeion	28.80	155.00	155.00	155.00	155.10
KNN	leuk	52.30	10.00	10.00	10.00	10.20
	arcene	33.50	5.00	5.00	5.00	4.80
	prostate	41.90	7.00	7.00	7.00	6.50
	lungcancer	275.60	21.00	21.00	21.00	21.10
	ovarian	12.70	12.00	12.00	12.00	11.90
	breastcancer	205.70	18.00	-	18.00	18.10
	dexter	41.30	15.00	15.00	15.00	15.00
	madelon	2.00	5.00	5.00	5.00	5.00
	sonar	2.30	9.00	9.00	9.00	8.50
	spambase	10.60	43.00	43.00	43.00	43.40
	spect	4.30	3.00	3.00	3.00	2.70
	semeion	28.80	184.00	184.00	184.00	184.40

2) *Compactness Metric:* Table VIII summarizes the number of selected features of SA-DCMB and other non-causal feature selection algorithms using the NB and KNN

TABLE IX
TIME (IN SECONDS) OF SA-DCMB AND NON-CAUSAL FEATURE SELECTION ALGORITHMS

Classifier	Dataset	FCBF	LASSO	QPFS	FSAE	SA-DCMB
NB	leuk	**0.33**	0.93	65.01	0.94	16.03
	arcene	**0.37**	1.54	120.45	1.31	15.75
	prostate	**0.31**	1.15	61.75	0.80	15.77
	lungcancer	**1.92**	3.12	466.97	2.11	141.43
	ovarian	**0.14**	0.84	36.79	0.39	17.57
	breastcancer	**2.53**	7.40	-	4.09	985.69
	dexter	**0.88**	4.91	347.89	3.79	13.74
	madelon	**0.15**	2.46	0.83	0.41	0.85
	sonar	**0.00**	0.13	0.05	0.01	0.01
	spambase	0.08	0.63	**0.07**	0.11	9.51
	spect	**0.00**	0.03	0.01	**0.00**	**0.00**
	semeion	**0.23**	2.03	0.35	0.81	515.32
KNN	leuk	**0.38**	0.99	65.01	0.97	25.45
	arcene	**0.37**	1.53	120.35	1.34	16.26
	prostate	**0.37**	1.11	94.56	0.84	21.96
	lungcancer	**1.98**	2.89	466.97	2.13	216.87
	ovarian	**0.16**	0.84	36.79	0.41	14.99
	breastcancer	**2.67**	6.89	-	4.13	926.89
	dexter	**0.98**	4.45	347.34	3.90	28.58
	madelon	**0.13**	2.37	0.89	0.42	0.33
	sonar	**0.00**	0.11	0.79	0.01	0.01
	spambase	**0.16**	0.86	0.28	0.28	11.99
	spect	**0.00**	0.02	0.01	**0.00**	**0.00**
	semeion	**0.22**	2.13	0.63	0.81	528.67

classifiers. Although LASSO, QPFS, and FSAE choose the same number of selected features with SA-DCMB in each dataset, the prediction accuracy of SA-DCMB is better than LASSO, QPFS, and FSAE on almost all the datasets, which means that SA-DCMB chooses more correct MB features. Furthermore, SA-DCMB selects fewer features and achieves higher accuracy than FCBF on high-dimensional and small-sized datasets (such as *leuk*, *arcene*, *prostate*, *lungcancer*, *ovarian*, *breastcancer*, and *dexter*), since the redundant features selected by FCBF lead to the overfitting of the predictive model on those datasets.

3) *Time Metric:* In Table IX, SA-DCMB is much faster than QPFS on most datasets. Especially, SA-DCMB is 25.3 times faster than QPFS on the *dexter* dataset, and 7.6 times faster than QPFS on the *arcene* dataset. Moreover, on the *breastcancer* dataset, QPFS runs for more than one day, while SA-DCMB runs for less than 1000 s. Since FCBF uses pairwise mutual information to calculate the relevancy between features and C, FCBF is significantly more efficient than SA-DCMB.

c) Statistical tests for verifying whether SA-DCMB is significantly better than other methods: In Section S-4 in the Supplementary Material, we adopt the Friedman test and Nemenyi test [39] to further compare the performance of SA-DCMB with that of its rivals.

VII. CONCLUSION

This article focuses on the two types of errors encountered by the existing causal feature selection methods: false positives (i.e., selecting false MB features) and false negatives (i.e., discarding true MB features). These two types of errors seriously deteriorate the performance of those existing methods. To tackle this problem, we propose a DCMB algorithm, which can efficiently and effectively correct the two types of errors simultaneously. Specifically, DCMB applies the AND rule to selectively delete false positives from the MB features currently selected, and then uses the OR rule to selectively recover false negatives from the features currently discarded. However, we cannot determine in advance how many features to select for correction to yield the best results. Thus, we further design the SA-DCMB algorithm to automatically search for the optimal number of selected features. Using benchmark BN datasets, our experimental results have validated the effectiveness of DCMB and demonstrated the rationale of the dual correction strategy of DCMB. In addition, SA-DCMB is extensively evaluated and compared with the state-of-the-art causal feature selection algorithms and well-established traditional feature selection methods on real-world datasets, and the results validate the effectiveness and superiority of SA-DCMB for feature selection.

In the future, we will further explore the following research issues in causal inference and causal feature selection. One direction is to extend DCMB for scalable learning of large-scale causal structures. The second direction is to use DCMB to tackle the problem of causal feature selection with multiple data sources, and the third direction is to extend DCMB to deal with a dataset with latent variables.

REFERENCES

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Amsterdam, The Netherlands: Elsevier, 2014.

[2] S. Yang, H. Wang, K. Yu, F. Cao, and X. Wu, “Towards efficient local causal structure learning,” *IEEE Trans. Big Data*, early access, Mar. 2, 2021, doi: 10.1109/TBDATA.2021.3062937.

[3] F. Zhu, J. Lu, A. Lin, and G. Zhang, “A Pareto-smoothing method for causal inference using generalized Pareto distribution,” *Neurocomputing*, vol. 378, pp. 142–152, Feb. 2020.

[4] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos, “Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: Algorithms and empirical evaluation,” *J. Mach. Learn. Res.*, vol. 11, pp. 171–234, Jan. 2010.

[5] I. Tsamardinos and C. F. Aliferis, “Towards principled feature selection: Relevancy, filters and wrappers,” in *Proc. Int. Workshop Artif. Intell. Statist.* Key West, FL, USA: Morgan Kaufmann Publishers, 2003, pp. 300–307.

[6] K. Yu, L. Liu, and J. Li, “A unified view of causal and non-causal feature selection,” *ACM Trans. Knowl. Discovery Data*, vol. 15, no. 4, pp. 1–46, Aug. 2021.

[7] I. Guyon, C. Aliferis, and A. Elisseeff, “Causal feature selection,” *Computational methods of feature selection*. London, U.K.: CRC Press, 2007, pp. 79–102.

[8] K. Yu *et al.*, “Causality-based feature selection: Methods and evaluations,” *ACM Comput. Surv.*, vol. 53, no. 5, pp. 1–36, 2020.

[9] J. M. Peña, R. Nilsson, and J. Björkegren, and J. Tegnér, “Towards scalable and data efficient learning of Markov boundaries,” *Int. J. Approx. Reasoning*, vol. 45, no. 2, pp. 211–232, Jul. 2007.

[10] S. Fu and M. C. Desmarais, “Fast Markov blanket discovery algorithm via local learning within single pass,” in *Proc. Conf. Can. Soc. Comput. Studies Intell.*, vol. 5032. Berlin, Germany: Springer, 2008, pp. 96–107.

[11] X. Wu, B. Jiang, K. Yu, C. Miao, and H. Chen, “Accurate Markov boundary discovery for causal feature selection,” *IEEE Trans. Cybern.*, vol. 50, no. 12, pp. 4983–4996, Dec. 2020.

[12] I. Tsamardinos, C. F. Aliferis, and A. Statnikov, “Time and sample efficient discovery of Markov blankets and direct causal relations,” in *Proc. 9th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2003, pp. 673–678.

[13] J. Li *et al.*, “Feature selection: A data perspective,” *ACM Comput. Surv.*, vol. 50, no. 6, p. 94, 2017.

[14] G. Brown, A. Pocock, M.-J. Zhao, and M. Lujan, “Conditional likelihood maximisation: A unifying framework for information theoretic feature selection,” *J. Mach. Learn. Res.*, vol. 13, pp. 27–66, Jun. 2012.

[15] J. R. Vergara and P. A. Estévez, “A review of feature selection methods based on mutual information,” *Neural Comput. Appl.*, vol. 24, no. 1, pp. 175–186, 2014.

[16] K. Yu, L. Liu, J. Li, W. Ding, and T. D. Le, “Multi-source causal feature selection,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 9, pp. 2240–2256, Sep. 2019.

[17] D. Koller and M. Sahami, "Toward optimal feature selection," in *Proc. Int. Conf. Int. Conf. Mach. Learn.* San Mateo, CA, USA: Morgan Kaufmann, 1996, pp. 284–292.

[18] A. A. Mastakouri, B. Schölkopf, and D. Janzing, "Necessary and sufficient conditions for causal feature selection in time series with latent common causes," in *Proc. Int. Conf. Mach. Learn.*, 2021, pp. 7502–7511.

[19] D. Margaritis and S. Thrun, "Bayesian network induction via local neighborhoods," in *Proc. Adv. Neural Inf. Process. Syst.*, 2000, pp. 505–511.

[20] S. Yaramakala and D. Margaritis, "Speculative Markov blanket discovery for optimal feature selection," in *Proc. 5th IEEE Int. Conf. Data Mining (ICDM)*, Nov. 2005, p. 4.

[21] I. Tsamardinos, C. F. Aliferis, A. R. Statnikov, and E. Statnikov, "Algorithms for large scale Markov blanket discovery," in *Proc. Int. Florida Artif. Intell. Res. Soc. Conf.*, vol. 2, 2003, pp. 376–380.

[22] A. Pocock, M. Luján, and G. Brown, "Informative priors for Markov blanket discovery," in *Proc. Int. Conf. Artif. Intell. Statist.*, 2012, pp. 905–913.

[23] G. Borboudakis and I. Tsamardinos, "Forward-backward selection with early dropping," *J. Mach. Learn. Res.*, vol. 20, no. 1, pp. 276–314, 2019.

[24] C. F. Aliferis, I. Tsamardinos, and A. Statnikov, "HITON: A novel Markov Blanket algorithm for optimal variable selection," in *Proc. Amer. Med. Informat. Assoc. Annu. Symp.*, 2003, p. 21.

[25] J. M. Pena, J. Björkegren, and J. Tegnér, "Scalable, efficient and correct learning of Markov boundaries under the faithfulness assumption," in *Proc. Eur. Conf. Symbolic Quant. Approaches Reasoning Uncertainty*, vol. 3571. Berlin, Germany: Springer, 2005, pp. 136–147.

[26] S. R. de Morais and A. Aussem, "A novel scalable and data efficient feature subset selection algorithm," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, vol. 5212. Berlin, Germany: Springer, 2008, pp. 298–312.

[27] T. Gao and Q. Ji, "Efficient Markov blanket discovery and its application," *IEEE Trans. Cybern.*, vol. 47, no. 5, pp. 1169–1179, May 2017.

[28] X. Wu, B. Jiang, K. Yu, and H. Chen, "Separation and recovery Markov boundary discovery and its application in EEG-based emotion recognition," *Inf. Sci.*, vol. 571, pp. 262–278, Sep. 2021.

[29] Z. Ling, K. Yu, H. Wang, L. Liu, W. Ding, and X. Wu, "BAMB: A balanced Markov blanket discovery approach to feature selection," *ACM Trans. Intell. Syst. Technol.*, vol. 10, no. 5, pp. 1–25, Sep. 2019.

[30] H. Wang, Z. Ling, K. Yu, and X. Wu, "Towards efficient and effective discovery of Markov blankets for feature selection," *Inf. Sci.*, vol. 509, pp. 227–242, Jan. 2020.

[31] I. Tsamardinos, L. E. Brown, and C. F. Aliferis, "The max-min hill-climbing Bayesian network structure learning algorithm," *Mach. Learn.*, vol. 65, no. 1, pp. 31–78, 2006.

[32] V. Černý, "Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm," *J. Optim. Theory Appl.*, vol. 45, no. 1, pp. 41–51, Jan. 1985.

[33] P. J. M. Van Laarhoven and E. H. L. Aarts, "Simulated annealing," in *Simulated Annealing: Theory and Applications*, vol. 37. Berlin, Germany: Springer, 1987, pp. 7–15.

[34] R. Tibshirani, "Regression shrinkage and selection via the lasso," *J. Roy. Statist. Soc., B, Methodol.*, vol. 58, no. 1, pp. 267–288, 1996.

[35] L. Yu and H. Liu, "Efficient feature selection via analysis of relevance and redundancy," *J. Mach. Learn. Res.*, vol. 5, pp. 1205–1224, Dec. 2004.

[36] I. Rodriguez-Lujan, R. Huerta, C. Elkan, and C. S. Cruz, "Quadratic programming feature selection," *J. Mach. Learn. Res.*, vol. 11, pp. 1491–1516, Mar. 2010.

[37] C. Lohrmann, P. Luukka, M. Jablonska-Sabuka, and T. Kauranne, "A combination of fuzzy similarity measures and fuzzy entropy measures for supervised feature selection," *Expert Syst. Appl.*, vol. 110, pp. 216–236, Nov. 2018.

[38] J. Li, L. Liu, and T. D. Le, *Practical Approaches to Causal Relationship Exploration*. Berlin, Germany: Springer, 2015.

[39] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

Xianjie Guo received the B.S. degree from Anhui Normal University, Wuhu, China, in 2018. He is currently pursuing the Ph.D. degree with the School of Computer Science and Information Engineering, Hefei University of Technology, Hefei, China.

His current research interests include causal discovery and machine learning.

Kui Yu (Member, IEEE) received the Ph.D. degree in computer science from the Hefei University of Technology, Hefei, China, in 2013.

From 2015 to 2018, he was a Research Fellow of computer science with the University of South Australia, Adelaide, SA, Australia. From 2013 to 2015, he was a Post-Doctoral Fellow with the School of Computing Science, Simon Fraser University, Burnaby, BC, Canada. He is currently a Professor with the School of Computer Science and Information Engineering, Hefei University of Technology. His main research interests include causal discovery and machine learning.

Lin Liu received the B.S. and M.S. degrees in electronic engineering from Xidian University, Xi'an, China, in 1991 and 1994, respectively, and the Ph.D. degree in computer systems engineering from the University of South Australia, Adelaide, SA, Australia.

She is currently an Associate Professor with the University of South Australia, Adelaide, SA, Australia, in 2006. Her research interests include data mining, causal inference, and bioinformatics.

Fuyuan Cao received the M.S. and Ph.D. degrees in computer science from Shanxi University, Taiyuan, China, in 2004 and 2010, respectively.

He is currently a Professor with the School of Computer and Information Technology, Shanxi University. His current research interests include machine learning and clustering analysis.

Jiuyong Li (Member, IEEE) received the Ph.D. degree in computer science from Griffith University, Brisbane, QLD, Australia, in 2002.

He is currently a Professor with the University of South Australia, Adelaide, SA, Australia. His main research interests include data mining, causal discovery, privacy preservation, and bioinformatics. His research work has been supported by seven Australian Research Council Discovery projects and many industry and government projects.