Received: 17 January 2020 | Revised: 23 February 2020 | Accepted: 14 March 2020

DOI: 10.1002/cpe.5778

RESEARCH ARTICLE

WILEY

A naive learning algorithm for class-bridge-decomposable multidimensional Bayesian network classifiers

Yali Lv[1,2] | Weixin Hu[1] | Jiye Liang[2] | Yuhua Qian[2] | Junzhong Miao[1]

[1]School of Information, Shanxi University of Finance and Economics, Taiyuan, China

[2]Key Laboratory of Computational Intelligence and Chinese Information Processing of Ministry of Education, Shanxi University, Taiyuan, China

Correspondence
Yali Lv, School of Information, Shanxi University of Finance and Economics, Taiyuan 030006, China.
Email: sxlvyali@126.com

Summary

Multidimensional Bayesian network classifier (MBC) has become a popular classification model because of their intuitive graphical representation ability among class variables. But learning MBC and performing multidimensional classification based on the MBC can be very computationally demanding. For the tractability of performing multidimensional classification, a class-bridge-decomposable (CB-decomposable) MBC model is proposed and it alleviates the computation complexity. But there are few works to efficiently and systematically learn the CB-decomposable MBC model. Thus, we focus on addressing a naive learning algorithm of CB-decomposable MBCs. Briefly, we learn the CB-decomposable MBC model by dividing it into three components: class subgraph, bridge subgraph, and feature subgraph. First, we analyze why the class subgraph can be learned based on general Bayesian network learning methods. Second, we give how to learn bridge subgraph based on information gain ratio. Third, to make the CB-decomposable MBC model effective and simple, we also study the learning and updating strategies of feature subgraph. Further, we propose the naive learning algorithm of the CB-decomposable MBC. Finally, by comparing with other methods on several benchmark datasets, experimental results illustrate that our naive learning algorithm not only has higher accuracies, lower learning, and classification times but also has simple and intuitive representation ability.

KEYWORDS

Bayesian networks, CB-decomposable multidimensional Bayesian network classifiers, information gain, machine learning, multidimensional classification, naive learning

1 | INTRODUCTION

Multidimensional classification[1] is an extension of the one-dimensional classification and is also a further development of multilabel classification. Its goal is to find a classifier that assigns a vector of class values by given feature vector. That is, for each instance having k features $\mathbf{f} = (f_1, f_2, \ldots, f_k)$, we need to learn a classifier model $\mathcal{M}$ that can assign the instance to a vector of d class values $\mathbf{c} = (c_1, c_2, \ldots, c_d)$. There are many application domains of multidimensional classification. For example, text categorization[2] can belong to multiple different topics, a scene can also have multiple classifications,[3] a gene can have multiple biological functions in bioinformatics,[4] malicious code[5,6] can be detected from multidimension, privacy-preserving, and so on.

There are many methods for multilabel classification.[7-11] These methods can be divided into two types: considering-relationships-among-labels method and not-considering-relationships-among-labels method. Binary relevance[10] is the classical not-considering-relationships method. In fact, considering the correlation among class variables has been proved to be the key to improve the accuracy of multilabel classification.[11] Label

Powerset is the classical considering-relationships method but it cannot predict unseen labelsets. These mentioned multilabel classification methods can be also used to solve the multidimensional classification problems.

However, Bayesian network (BN) classifiers[12] not only can represent relationships among variables and predict unseen class value but also have good interpretability, it is widely used in many supervised classification fields. In order to solve multidimensional classification better, the multidimensional Bayesian network classifier (MBC)[1] is proposed by introducing the concept of multidimensionality into Bayesian network classifiers. So an MBC structure $\mathcal{G}$ includes three subgraphs: class subgraph $\mathcal{G}_C$, bridge subgraph $\mathcal{G}_B$, and feature subgraph $\mathcal{G}_F$. Where $\mathcal{G}_C$ and $\mathcal{G}_F$ may be empty, directed tree, polytree, and directed acyclic graphs (DAG). Thus, there are several different families of MBCs that are caused by different graphical structures of the class subgraph and feature subgraph, such as tree-tree MBCs,[1] polytree-polytree MBCs,[13] DAG-DAG MBCs,[30] and so on.

In order to learn a better MBC structure, researchers have developed many methods. Van der Gaag et al[1] proposed a tree-tree MBC learning method, which shows that the complexity of algorithm is polynomial in the number of variables involved. Specifically, for class subgraph and feature subgraph, they first learned the maximum weighted generation trees based on minimum description length score function. And then they used the general feature selection method to learn the bridge subgraph. Furthermore, they expanded to learn the polytree-polytree MBCs[13] and addressed the bound of tree width of the MBC DAG $\mathcal{G}$, the bound meets

$$\text{treewidth}(\mathcal{G}) \leq \text{treewidth}(\mathcal{G}_F) + |V_C|.$$

This means that an MBC $\mathcal{G}$ could conduct multidimensional classification in polynomial time if the sum of the feature subgraph's treewidth($\mathcal{G}_F$) and the number of class variables $|V_C|$ is bounded. Qazi et al[14] learned a DAG-empty MBC for automated heart wall motion abnormality detection, and they learned class subgraph using general BN learning method and took prior domain knowledge to learn bridge subgraph by adding dependence relationship between each class and features. Rodriguez et al[30] presented a k dependence Bayesian MBC based on multiobjective evolutionary algorithm NSGA-II that can be used to many fields.[15-18] While k is enough large, in fact, the k dependence Bayesian MBC becomes a DAG-DAG MBC.

The above-mentioned learning methods are addressed according to the type of class and feature subgraphs. However, the most probable explanation (MPE) must be computed in multidimensional classification based on BN. But these methods do not explicitly solve the inference complexity based on BN classifier model. So Benjumeda et al[19] researched the tractability of MPE in MBCs. They provided upper bounds for the complexity of MPE and marginal of class variables conditioned to an instantiation of all feature variables. Moreover, during the learning process, they used these bounds to propose efficient rules to bound model complexity. As well, Bielza et al[20] proposed class-bridge decomposable MBC (CB-decomposable MBC), which can be decomposed into r maximal connected components. In this work, the theoretical results are provided in the basic MBC and the CB-decomposable MBC. In Reference [21], based on a wrapper greedy forward selection method, Borchani et al proposed a novel learning algorithm of CB-decomposable MBCs. Based on selective naive Bayes, they first learned bridge subgraph in which there is no common feature variables for two different class variables. And then they learned feature subgraph by adding randomly edges until classification accuracy does not improve. Next, they merged maximal connected components by learning class subgraph. Finally, experimental results showed that the computations of MPE of CB-decomposable MBCs are alleviated comparing to general MBCs.

However, this learning algorithm[21] focuses on demonstrating the usefulness of CB-decomposable MBCs and the alleviated computations of MPE by experimental study. Neither time complexity of learning algorithm nor the experimental comparison with the popular DAG-DAG CB-decomposable MBCs, which have the strong representation ability, have been researched efficiently and systematically. Therefore, in this article, we address a naive learning algorithm for a DAG-DAG CB-decomposable MBC model. Where class subgraph is learned based on general BN learning algorithms, bridge subgraph is learned only once, feature subgraph needs to be learned or updated based on its bridge subgraph and make its graphical representation simple, which reduces the learning complexity and also guarantees the MBC to be still tractable.

The rest of this article is arranged as follows: Section 2 reviews some preliminaries. In Section 3, we propose the naive learning algorithm of CB-decomposable MBCs. In Section 4, experiments demonstrate the performance of our proposed algorithm by comparing with other methods. Finally, Section 5 concludes this article.

2 | PRELIMINARIES

In this part, we introduce some concepts of multidimensional Bayesian networks classifiers (MBCs) and the CB-decomposable MBCs.

2.1 | MBCs

Essentially, an MBC is a Bayesian network (BN) specially designed to solve classification problems including multiple class variables. Given all feature values, we need to learn the MPE of multiple class variables. Its definition is given as follows.

Definition 1 (Multidimensional Bayesian networks classifier, MBC[1,20]). An MBC model can be formally expressed as $\mathcal{M} = (\mathcal{G}, \Theta)$, $\mathcal{G} = (V, E)$ is a DAG, where the set of variables $V = \{V_1, V_2, \dots, V_n\}$ is partitioned into two sets: the set of class variables $V_C = \{C_1, C_2, \dots, C_d\}, d \geq 1$, and the set of feature variables $V_F = \{F_1, F_2, \dots, F_k\}, k \geq 1$, and $k + d = n$. Where n is the number of all variables in an MBC model. E is partitioned into three sets:

- E_C. $E_C \subseteq V_C \times V_C$ is the set of the edges between class variables. $\mathcal{G}_C = (V_C, E_C)$ is the corresponding class subgraph.
- E_F. $E_F \subseteq V_F \times V_F$ is the set of the edges between feature variables. $\mathcal{G}_F = (V_F, E_F)$ is the corresponding feature subgraph.
- E_B. $E_B \subseteq V_C \times V_F$ is the set of the edges from class variables to feature variables. $\mathcal{G}_B = (V, E_B)$ is the corresponding bridge subgraph, where $V = V_C \cup V_F$.

Θ is the set of parameters $\theta_{v|\Pi(v)} = \Pr(v|\Pi(v))$, where $\Pi(V)$ is the set of parents of variable V in the graphical structure $\mathcal{G}$. $\Pi(v)$ is a combination value of all parents of V, that is, $\Pi(v)$ is a value of $\Pi(V)$. An MBC as a special BN model, it also can be described as a joint probability distribution over V, that is,

$$\Pr(V_1, V_2, ..., V_n) = \prod_{i=1}^{n} \Pr(V_i|\Pi(V_i)). \tag{1}$$

A simple example of DAG-DAG MBCs and its three subgraphs are shown in Figure 1. There are four class variables and seven feature variables in the MBC model.

2.2 | CB-decomposable MBCs

To alleviate the MPE computational complexity, References [20] and [21] consider the CB-decomposable MBCs and show that, in a CB-decomposable MBC, the maximization problem for MPE computation can be transformed into r maximization problems that conducted in lower dimensional spaces. In addition, comparing with the whole large and complex MBCs, more insight about the domain and better interpretability can be provided using a CB-decomposable MBC model.

Definition 2 (CB-decomposable MBCs[20,21]). Suppose we have an MBC model where $\mathcal{G}_C$ is its class subgraph and $\mathcal{G}_B$ is its bridge subgraph. We say that the MBC is class-bridge decomposable, denoted CB-decomposable, if

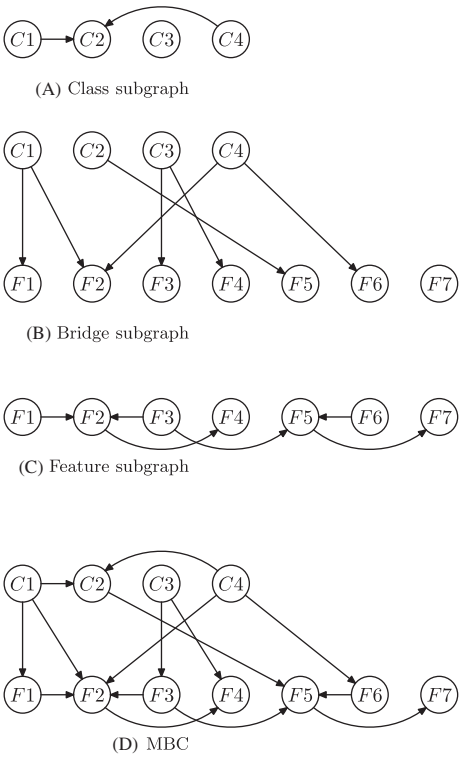


FIGURE 1 An example of the MBC and its three subgraphs. (A) Class subgraph. (B) Bridge subgraph. (C) Feature subgraph. (D) The MBC

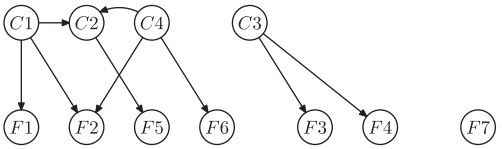


FIGURE 2 The CB-decomposable MBC of the model in Figure 1

- $\mathcal{G}_C \cup \mathcal{G}_B$ can be decomposed as $\mathcal{G}_C \cup \mathcal{G}_B = \cup_{i=1}^{r}(\mathcal{G}_C^i \cup \mathcal{G}_B^i)$, where $\mathcal{G}_C^i \cup \mathcal{G}_B^i (i = 1, ..., r)$ are the r maximal connected components.
- $\mathrm{Ch}(V_{\mathcal{G}_C^i}) \cap Ch(V_{\mathcal{G}_C^j}) = \emptyset$ with $i, j = 1, \ldots r$ and $i \neq j$, where $\mathrm{Ch}(V_{\mathcal{G}_C^i})$ denotes the children of all class variables in ith connected component. This means that there are no common children for two class variables in different connected components.

To sum up, CB-decomposable MBCs can solve the MPE in each component and avoid the complexity of higher dimensional computing. The CB-decomposable MBC of the model in Figure 1 is shown in Figure 2. Here it is decomposed into two maximal connected components and one outlier feature variable.

3 | NAIVE LEARNING OF CB-DECOMPOSABLE MBCS

In this section, we first describe the idea of the CB-decomposable MBC's naive learning and then propose two key learning strategies. Furthermore, the pseudocode description of naive learning and its complexity analysis are given in detail.

3.1 | The basic idea

Since the multidimensional classification using an MBC model aims at getting the MPE $\mathbf{c}^*$ of the class variables given an instantiation of the feature variables $\mathbf{f}$, which is given by

$$\mathbf{c}^* = \arg\max_{\mathbf{c}\in\Omega_C} \Pr(\mathbf{c}|\mathbf{f}) = \arg\max_{\mathbf{c}\in\Omega_C} c. \tag{2}$$

That is, making $\Pr(\mathbf{c}|\mathbf{f})$ maximal equals making $\Pr(\mathbf{c}, \mathbf{f})$ maximal. Where Ω_C denotes the state space of class variables C or V_C. Thus, we can learn CB-decomposable MBCs by general methods of learning BN model and here we use Bayesian information criterion (BIC) score function to measure CB-decomposable MBCs. The BIC score function is expressed as

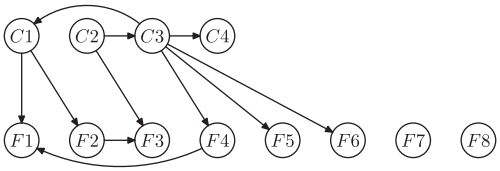
$$\mathrm{BIC}(\mathcal{M}) = \sum_{i=1}^{n} \mathrm{BIC}(V_i|\Pi_{V_i}). \tag{3}$$

From Equation (3), we have the score function is decomposable according to each variable or node; moreover, the BIC of each node is only related with its parent nodes. Thus

- for class subgraph $\mathcal{G}_C$, since feature variables can only be children of class variables in an MBC, they are not related with the BIC scores of each class variable. Thus, we can directly learn $\mathcal{G}_C$ based on general BN structure learning methods. Which does not change the BIC scores of class subgraph. Clearly, class subgraph does not need to be updated based on other subgraphs.
- for bridge subgraph $\mathcal{G}_B$, in order to make the MBC still be decomposable and the update time complexity be alleviated, we learn the most relative several feature variables for each class variable. Also we do not allow that there are any common feature variables between each pair class variables. Note that this process can be learned only once and not be updated any more later. So learning a good bridge subgraph is rather important for multidimensional classification based on CB-decomposable MBCs.
- for feature subgraph $\mathcal{G}_F$, we can still learn the directed edges based on general BN learning algorithms, but we need to verify whether the learned edges among feature variables could be added in the end by considering its corresponding class parent in bridge subgraph. Which makes the learned $\mathcal{G}_F$ more simple than that of being learned only using general BN learning ones.

We call this learning process as a naive learning algorithm and the learned CB-decomposable MBC models are also called naive CB-decomposable MBCs. The formal definition of our naive CB-decomposable MBC is as follows.

FIGURE 3 A simple example of naive CB-decomposable MBCs



Definition 3 (Naive CB-decomposable MBC). A naive CB-decomposable MBC, whose feature variables have no or only one class variable as their class parent node, can be written as $\mathcal{M}^{naive} = (\mathcal{G}^{naive}, \Theta^{naive})$. Similarly, its $\mathcal{G}^{naive}$ includes

- Class subgraph $\mathcal{G}_C^{naive} = (V_C^{naive}, E_C^{naive})$, where E_C^{naive} only contains directed edges between class variables.
- Bridge subgraph $\mathcal{G}_B^{naive} = (V_B^{naive}, E_B^{naive})$, where E_B^{naive} only contains directed edges from class variables to feature variables, each feature variable has no or only one class parent.
- Feature subgraph $\mathcal{G}_F^{naive} = (V_F^{naive}, E_F^{naive})$. E_F^{naive} only contains directed edge between feature variables.

A simple example of our naive CB-decomposable MBC model is shown in Figure 3. There are four class variables and eight feature variables. Therefore, we mainly study the naive learning strategies of bridge subgraph $\mathcal{G}_B^{naive}$ and feature subgraph $\mathcal{G}_F^{naive}$ in the following subsections.

3.2 | $\mathcal{G}_B^{naive}$ learning based on information gain ratio

The bridge subgraph of our naive CB-decomposable MBC aims at learning the directed edges between class and feature variables. For each class variable, we need to find the most relative features and guarantee there are no common features between class variables. We know that famous decision tree can be used to conduct classification task well, where information entropy, information gain, and information gain ratio are often used to measure the importance of features for a class. Because information gain ratio not only considers the degree of information gain but also reflects the costs of splitting information entropy (Split entropy), here we adopt information gain ratio to find the most relative feature variables for each class variable. The related concepts of information gain ratio are formally described as follows.

$$\text{Entropy(C)} = -\sum \Pr(c) \cdot \log_2 \Pr(c), \tag{4}$$

where Entropy(C) measures the amount of information of class variable C in dataset. The smaller the Entropy(C) is, the purer the distribution of the data to C is. $\Pr(c)$ is the probability of class variable C taking a certain value in dataset.

$$\begin{aligned} &\text{Information gain(C,F)} \\ &= \text{Entropy(C)} - \text{Entropy}_F(C) \\ &= \text{Entropy(C)} - \sum_{i=1}^{p} \frac{S_{F_i}}{S_C} \cdot \mathit{Entropy}(F_i). \end{aligned} \tag{5}$$

Where $\text{Entropy}_F(C)$ means the entropy of sub datasets that are divided into by feature variable F. Assume that F has p values and the dataset has been divided into p sub datasets, so $\text{Entropy}_F(C) = \sum_{i=1}^{p} \frac{S_{F_i}}{S_C} \cdot \mathit{Entropy}(F_i)$. S_{F_i} denotes the instance number of the ith value of F in data. S_C denotes the instance number of the class variable C in data. $\text{Entropy}(F_i)$ means the amount of information of class variables C in ith sub dataset, where F takes the ith value. The bigger the Information gain(C,F) is, the purer the distribution of the p sub data to C according to feature variable F is.

In order to avoid that the result is biased to the feature variable with many values, in this article, we use information gain ratio, which not only considers the information gain but also uses the cost of split information by F, to measure the classification abilities. So

$$\text{Information gain ratio(C,F)} = \frac{\text{Information gain(C,F)}}{\text{Split entropy(F)}} \tag{6}$$

Where

$$\text{Split entropy(F)} = -\sum_{i=1}^{p} \frac{S_{F_i}}{S_C} \cdot \log_2 \frac{S_{F_i}}{S_C} \tag{7}$$

The bigger the ratio is, the more suitable it is used to classify the class.

Specifically, for each class variable, as decision tree based on information gain, we select features which have higher information gain ratio for q layers. As well, to ensure the performance better, the sample data with the same sizes are trained u times and we random sample 30% dataset to construct decision trees at each time. And then we union all features in q layers as the current class variable's children. We loop until the features of all class variables are selected out. Note that the layer number q is also as the stop condition of iteration during learning decision tree. For the layer number q, it should neither be too big (increasing complexity and overfitting) nor too small (having not enough features to represent its class). It can be given according to each class variable that has as many features as possible to describe it.

Furthermore, for all class variables, there must be common feature variables in the results. However, good separability means greatly reducing computational complexity of classification based on MBC models. In order to ensure that the whole graph has r components, we only keep the common feature variables in the class component with the highest average information gain ratio and eliminate them from other class components. Which guarantees each component has the most relative and unique feature variables and these feature variables' values have direct effect on the current class variable for classification task.

In brief, differing from building a pure C4.5 decision tree, learning the relative feature variables of each class variable in bridge subgraph does not need to build the whole decision tree and it needs to reach the given layer number, which reduces the computational complexity and also avoids over fitting to some extent. In addition, to ensure CB-decomposable, for all class variables, we cannot allow there are common feature variables between any two class variables.

3.3 | $\mathcal{G}_F^{naive}$ learning and updating

For feature variables, their parents may be class variables or feature variables. So, during learning feature subgraph $\mathcal{G}_F^{naive}$ according to BIC score function, we need to consider class variables or update $\mathcal{G}_F^{naive}$ by $\mathcal{G}_B^{naive}$.

In our naive CB-decomposable MBC model, the class parent node of some feature variables maybe only one or no. Thus we give the special strategies of learning and updating $\mathcal{G}_F^{naive}$. First, we learn $\mathcal{G}_F^{naive}$ based on general BN learning methods. For each feature variable V_{F_i}, if it adds one or multiparents $\Pi(V_{F_i})$ during learning $\mathcal{G}_F^{naive}$, then we need to update these edges using its class parent node $\Pi(V_{F_i}^B)$ in $\mathcal{G}_B^{naive}$. That is, if

$$\mathrm{BIC}(V_{F_i}|(\Pi(V_{F_i}) \cup \Pi(V_{F_i}^B))) > \mathrm{BIC}(V_{F_i}|\Pi(V_{F_i}^B)),$$

then we finally add these edges as parents of V_{F_i} and do not add them otherwise.

For example, for feature F_1 in Figure 3, we learn $\mathrm{BIC}(F_1|F_4) > BIC(F_1)$ in $\mathcal{G}_F^{naive}$ by general BN learning methods, and then we add temporarily $F_4 \rightarrow F_1$. Next, assume that we have $C_1 \rightarrow F_1$ showed in Figure 3, which is learned by the method in Section 3.2. Furthermore, we also need to judge $\mathrm{BIC}(F_1|F_4, C_1) > BIC(F_1|C_1)$, if the inequality is ok, we retain the edge $F_4 \rightarrow F_1$ and delete it otherwise. This makes the relationship among feature variables more reasonable and the feature subgraph simpler.

3.4 | The whole naive learning algorithm

The goal of the naive learning algorithm of a CB-decomposable MBC model is to ensure its simplicity, accuracy, and tractability, the learning process is divided into three steps. First, the class subgraph $\mathcal{G}_C^{naive}$ is learned based on general BN learning methods. Second, using the proposed strategies in Section 3.2, we learn the bridge subgraph $\mathcal{G}_B^{naive}$ structure. Third, we learn and update feature subgraph $\mathcal{G}_F^{naive}$ based on the given strategies in Section 3.3. In a conclusion, the pseudocode of our naive learning algorithm is described in Algorithm 1.

For the time complexity of Algorithm 1, we also analyze them step by step.

In terms of learning class subgraph $\mathcal{G}_C^{naive}$ in step 1 (Line 2), different BN learning methods have different time complexities. If we use K2 algorithm[22] as the general BN learning method, the time complexity is $O(m \cdot t_C^2 \cdot d^2 \cdot |\mathrm{dom}_{max}(C)|)$, where m is the number of sample data, t_C is the maximum number of parent nodes of class variables, d is the number of class variables, and $|\mathrm{dom}_{max}(C)|$ is the maximum number of domain value of class variables.

In step 2 (Lines 4 to 17), for each class variable, the time complexity of learning its children from feature variables once is similar to learning a decision tree, it is $O(m \cdot k^2)$,[23,24] k is the number of feature variables. Since there are d class variables and we learn u times on the sample data with the equal sizes, the time complexity of the first **for** loop from Lines 4 to 10 is $O(d \cdot u \cdot m \cdot k^2)$. For the second **for** loop from Lines 11 to 15, there are $\binom{d}{2}$ pairs of class variables, it has $O(d^2)$. Thus, the time complexity of learning bridge subgraph $\mathcal{G}_B^{naive}$ is $O(d \cdot u \cdot m \cdot k^2) + O(d^2)$.

For learning feature subgraph $\mathcal{G}_F^{naive}$, if we use K2 algorithm to learn the DAG among features in Line 20, the time complexity is $O(m \cdot t_F^2 \cdot k^2 \cdot |\mathrm{dom}_{max}(F)|)$, t_F is the maximum number of parent nodes of feature variables, and $|\mathrm{dom}_{max}(F)|$ is the maximum number of domain value of feature variables. Further we need to verify the final edges for k feature variables by **for** loop from Lines 19 to 24, so the time complexity of learning feature subgraph $\mathcal{G}_F^{naive}$ is $O(m \cdot t_F^2 \cdot k^3 \cdot |\mathrm{dom}_{max}(F)|)$.

Algorithm 1. Naive learning algorithm of CB-decomposable MBCs

Input: Dataset $\mathcal{D}$, class variables V_C, the number of class variables d,learning times for each class u, the layer number of decision tree q;
Output: A CB-decomposable MBC $\mathcal{G}$

1: **Learning class subgraph** $\mathcal{G}_C^{naive}$:
2: learning $\mathcal{G}_C^{naive}$ based on general BN methods;
3: **Learning bridge subgraph** $\mathcal{G}_B^{naive}$:
4: **for** $i = 1$ to d **do**
5: $F_{C_i} = \emptyset$, where F_{C_i} denotes the feature set of ith class variable;
6: **for** j to u **do**
7: learning q layer decision tree based on information gain ratio and obtain $F_{C_{ij}}$;
8: $F_{C_i} \leftarrow F_{C_i} \cup F_{C_{ij}}$;
9: **end for**
10: **end for**
11: **for** any two class variables **do**
12: **if** there are common features **then**
13: delete the common features with lower average information gain ratio from its corresponding F_{C_i};
14: **end if**
15: **end for**
16: add edges from each class V_{C_i} to its each feature in F_{C_i};
17: return $\mathcal{G}_B^{naive}$;
18: **Learning feature subgraph** $\mathcal{G}_F^{naive}$:
19: **for** each feature V_{F_i} **do**
20: finding its parent set $\Pi(V_{F_i})$ that has been learned based on general BN methods
21: **if** $\mathrm{BIC}(V_{F_i}|\Pi(V_{F_i}) \cup \Pi(V_{F_i}^B)) > \mathrm{BIC}(V_{F_i}|\Pi(V_{F_i}^B))$ **then**
22: add these edges in $\Pi(V_{F_i})$ as parents of V_{F_i} in the end;
23: **end if**
24: **end for**
25: return $\mathcal{G}_F^{naive}$;
26: $\mathcal{G} = \mathcal{G}_C^{naive} \cup \mathcal{G}_B^{naive} \cup \mathcal{G}_F^{naive}$;
27: return $\mathcal{G}$;

In addition, there are three parts that can be learned in parallel by our Algorithm 1, including 1) class subgraph and bridge subgraph can be learned in parallel; 2) features of each class in bridge subgraph can be learned (from Lines 4 to 10) in parallel according to each class variable; 3) feature subgraph can be updated (from Lines 19 to 24) in parallel according to each feature variable. Which can speed up the execution of the algorithm.

4 | EXPERIMENTS

In this section, we verify the performance of our proposed learning algorithm from two aspects: the learned CB-decomposable MBC model and the classification accuracy based on this model. First, we introduce the experiment settings, dataset, and the adopted evaluation metrics for this experiment. Second, we analyze our experimental results in different sample sizes. Next, we show the result analysis of our learned CB-decomposable MBC by comparing with several existing BN-based learning methods. Finally, we demonstrate the classification accuracy according to global accuracy and mean accuracy by comparing with other non-BN multidimensional classification methods.

4.1 | Experiment settings and datasets

The experiment environment includes a PC with Mac, a 1.8 GHz Intel Core i5 CPU, and 8.00 GB main memory. All codes are completed in python 3.7 with the compiler PyCharm.

In this experiment, we use six synthetic datasets generated from the corresponding benchmark BNs, which can be found at http://www.bnlearn.com/bnrepository/. We sample three different instance sizes and they are 1000, 10 000, and 30 000. The details of six benchmark BNs are shown in Table 1, including BN name, instance sizes m, the number of nodes $|V|$, the number of directed edges $|E|$, the number of parameters $|P|$, and the maximum number of parent nodes t.

From Table 1, we can have that the number of nodes is from 8 to 76, the number of directed edges is from 8 to 112, the number of parameters is from 18 to 10 083, and the maximum number of parent nodes is from 2 to 7. This covers several different cases.

TABLE 1 Details of six benchmark BNs

Name	m	$\|V\|$	$\|E\|$	$\|P\|$	t
Asia	1000				
	10 000	8	8	18	2
	30 000				
Sachs	1000				
	10 000	11	17	178	3
	30 000				
Alarm	1000				
	10 000	37	46	509	4
	30 000				
Insurance	1000				
	10 000	27	52	984	3
	30 000				
Water	1000				
	10 000	32	66	10 083	5
	30 000				
Win95pts	1000				
	10 000	76	112	574	7
	30 000				

4.2 | Evaluation metrics

Evaluating the proposed learning algorithm can be divided into two parts. One part is for the performance of the learned CB-decomposable MBC model, we use seven metrics, M_1, M_2, M_3, M_4, *precision*, *recall*, and *times* to test by comparing with the corresponding benchmark BN structures, where the first four are proposed in Reference [21]. The other part is for the performance of classification base on learned CB-decomposable MBC model, we use three metrics: the global accuracy acc_G, the mean accuracy acc_M, and *times* to measure, where the first two are proposed in Reference [19]. These evaluation metrics are as follows in details.

1. M_1. Percentage of edges that are correctly present in the learned CB-decomposable MBC structure.
2. M_2. Percentage of edges incorrectly present in the learned CB-decomposable MBC structure. The smaller the value is, the better the model is.
3. M_3. Percentage of opposite direction edges in the learned CB-decomposable MBC structure. The smaller the value is, the better the model is.
4. M_4. Percentage of absent edges in the learned CB-decomposable MBC structure. The smaller the value is, the better the model is.
5. *precision*. $precision = \frac{M_1}{M_1+M_2}$.
6. *recall*. $recall = \frac{M_1}{M_1+M_4}$.
7. acc_M. It means the mean accuracy of classifiers which averages the accuracy values of all class variables individually. It can be described as

$$acc_M = \text{Mean accuracy}(M) = \frac{1}{d}\sum_{j=1}^{d}\frac{1}{m}\sum_{i=1}^{m} f(\hat{c}_{ij} = c_{ij}), \tag{8}$$

   where m is the number of samples, d is the number of class variables, $\hat{c}_{ij}$ and c_{ij} express the predicted class value for variable C_j in instance i and its true value, respectively. If $\hat{c}_{ij} = c_{ij}$, $f(\text{True}) = 1$, and $f(\text{False}) = 0$ otherwise.
8. acc_G. It means the global accuracy and is given by

$$acc_G = \text{Global accuracy}(G) = \frac{1}{m}\sum_{i=1}^{m} f(\hat{c}_i = c_i), \tag{9}$$

   where m, d, and f as acc_M. It can be seen that acc_G is more strict than acc_M.

9. *times(s)*. It means the model's learning times or classification times based on the learned CB-decomposable MBC model. *s* denotes the running time is measured using second.
10. ±. It represents the standard deviation of multiple results in experiments. The smaller the standard deviation is, the more stable the experimental results are.

4.3 | Results analysis in different sample sizes

To verity the classification performance, we first address the sample size which can make the MBC stable. Therefore, the experiments are conducted on six BNs dataset by our proposed algorithm in three different sample sizes, such as 1000, 10 000, and 30 000. There are 18 (3*6) cases and we conduct five times in each cases. Note that when the node orders are known, we take K2 algorithm as the general BN learning method; when they are unknown, we adopt hill climbing (HC) algorithm.

The experimental average results are shown in Table 2 based on K2 algorithm and in Table 3 based on HC algorithm, respectively.

From Tables 2 and 3, we can have the following conclusions.

1. The classification accuracies of acc_G and acc_M all increase with the increasing sample sizes except for Asia BN in Table 2 and in Table 3. Their standard deviations decrease with the increase in sample size, except for acc_G and acc_M of Asia in Table 2, acc_G of Alarm and Win95pts in Table 2, acc_G of Sachs and Alarm in Table 3. This shows that the algorithm tends to be more accurate and stable with the increase of sample size.
2. For measuring the learned CB-decomposable MBC based on K2 algorithm or HC algorithm, in most cases, the learning performance in bigger sample size is better than that of in smaller sample size by M_1, M_2, M_3, M_4. Moreover, M_3 in Table 2 are all zero, because the node orders of six BNs are given, it is impossible to learn the reverse edges.
3. Furthermore, we can conclude that the learned CB-decomposable MBC model, their classification accuracies, and standard deviations have smaller change from 10 000 to 30 000 than that of from 3000 to 10 000 in most cases. This shows that the results are relative stable in 10 000. Thus, in the following subsection, we will verify our proposed algorithm in 10 000 sample size.

TABLE 2 The experimental results in three different sample data based on K2 algorithm

		The learned MBC model (Ave 5 times)			Classification accuracy (Ave 5 times)		
Name	*m*	M_1	M_2	M_3	M_4	acc_G	acc_M
Asia	1000	87.50±0.00	25.00±0.00	0.00±0.00	12.50±0.000	68.70±1.71	89.62±0.61
	10 000	92.50±0.47	25.00±0.00	0.00±0.00	7.51±0.47	66.43±0.34	88.44±0.15
	30 000	100.00±0.00	25.00±0.00	0.00±0.00	0.00±0.00	66.51±0.72	88.75±0.32
Sachs	1000	80.00±0.94	9.41±0.47	0.00±0.00	20.00±0.94	63.00±1.53	80.25±1.14
	10 000	94.12±0.82	11.77±0.82	0.00±0.00	5.88±0.81	67.70±1.30	81.98±0.41
	30 000	94.12±0.82	11.77±0.82	0.00±0.00	5.88±0.81	67.70±1.20	82.42±0.25
Alarm	1000	62.17±0.49	40.87±0.75	0.00±0.00	37.83±0.49	65.82±0.44	93.41±1.24
	10 000	78.70±0.98	32.61±0.00	0.00±0.00	20.44±0.80	68.23±0.51	95.53±0.25
	30 000	80.80±0.60	31.74±0.01	0.00±0.00	19.13±0.49	68.41±0.10	95.84±0.21
Insurance	1000	19.23±0.90	10.38±0.80	0.00±0.00	79.23±0.71	36.40±1.10	65.00±2.51
	10 000	25.00±0.00	9.62±0.00	0.00±0.00	75.00±0.00	38.51±0.71	67.33±0.42
	30 000	26.54±0.49	9.23±0.02	0.00±0.00	74.62±0.82	38.75±0.34	67.51±0.12
WaterWater	1000	11.81±0.49	10.60±0.71	0.00±0.00	89.39±0.06	6.41±0.10	54.14±1.14
	10 000	19.39±0.40	11.51±0.80	0.00±0.00	80.90±0.49	7.82±0.01	56.12±0.31
	30 000	20.30±0.34	10.90±0.02	0.00±0.00	80.00±0.71	8.21±0.01	56.31±0.20
Win95pts	1000	50.00±0.00	34.82±0.00	0.00±0.00	57.67±0.04	75.70±0.08	97.65±2.10
	10 000	61.60±0.02	33.92±0.49	0.00±0.00	38.39±0.06	78.81±0.02	98.32±1.14
	30 000	63.21±0.47	58.92±0.01	0.00±0.00	38.75±0.00	79.14±0.04	98.61±0.31

TABLE 3 The experimental results in three different sample data based on HC algorithm

		The learned MBC model (Ave 5 times)			Classification accuracy (Ave 5 times)		
Name	m	M_1	M_2	M_3	M_4	acc_G	acc_M
Asia	1000	37.50±0.81	50.00±0.00	20.00±1.24	42.50±0.94	63.71±0.60	87.61±0.90
	10 000	50.00±0.82	45.00±0.47	7.50±0.47	30.00±0.94	65.95±0.40	87.15±0.80
	30 000	52.50±0.47	50.00±0.47	25.00±0.82	30.00±0.47	66.50±0.01	88.30±0.11
Sachs	1000	41.18±0.00	21.18±1.24	3.52±0.47	55.29±0.47	64.00±0.50	79.80±2.11
	10 000	47.05±0.81	41.17±0.81	5.88±0.00	47.05±0.81	64.11±0.40	80.59±0.50
	30 000	49.41±0.47	38.82±1.24	5.88±0.00	44.70±0.47	64.52±0.70	80.70±0.40
Alarm	1000	23.04±0.50	38.26±1.50	6.52±0.00	70.87±0.50	53.03±0.51	93.80±0.90
	10 000	26.95±0.47	36.52±0.00	7.39±0.94	64.78±0.01	65.12±0.10	94.33±0.21
	30 000	27.83±0.49	35.65±0.00	9.56±0.49	62.61±0.00	65.81±0.14	94.50±0.14
Insurance	1000	6.54±0.49	11.54±0.89	3.85±0.40	93.46±0.50	29.80±0.41	55.50±1.04
	10 000	11.92±0.75	14.23±0.80	0.39±0.40	86.92±0.40	35.81±0.35	61.35±0.25
	30 000	13.08±0.07	13.85±0.49	0.38±0.03	88.08±0.03	36.40±0.31	62.50±0.00
Water	1000	6.67±0.80	13.33±0.75	5.15±0.49	88.18±0.98	5.52±0.14	55.20±0.84
	10 000	9.70±0.79	11.51±0.80	5.40±0.49	87.27±0.49	6.64±0.05	56.85±0.14
	30 000	10.60±0.00	10.90±0.04	5.80±0.79	86.06±0.01	7.10±0.01	57.10±0.13
Win95pts	1000	11.61±0.47	33.93±0.00	5.00±0.03	83.57±0.00	72.40±0.31	96.40±0.45
	10 000	14.29±0.49	37.32±0.05	6.43±0.02	80.71±0.49	75.32±0.24	97.92±0.01
	30 000	15.00±0.00	35.89±0.47	6.25±0.00	80.17±0.82	75.82±0.00	98.11±0.00

4.4 | Comparison result analysis based on serval BN learning methods

In this subsection, by comparing our proposed naive learning method with several BN learning methods, we demonstrate the experimental results of the learned CB-decomposable MBC models and the classification based on these models. Before comparison, there are two keys that need us to describe in detail.

- Selecting class variables. In benchmark BNs, we first need to select out some class variables. In general, as a class variable V_{C_i} in a CB-decomposable MBC model, its parents $\Pi(V_{C_i})$ are only considered as class variables, not as feature variables. In order to make the model simple, we also consider that the common children of V_{C_i} and $V_C \setminus V_{C_i}$ are as less as possible. According to the two rules, the class variables of several BNs in Table 1 are shown in Table 4, including the number of class variables d and index of class variables I_d.
- Sampling training data and test data. For the sample data that have been sampled from six benchmark BNs in Table 1, we divide them into training data (90%) and test data (10%). We first use our naive learning algorithm to learn the CB-decomposable MBC model from training data and then to classify for test data. Moreover, for each m of benchmark BNs, we randomly sample five times ($u = 5$) to conduct the experiments.

TABLE 4 The details of class variables on six BNs dataset

Name	d	I_d
Asia	3	[0,1,2]
Sachs	2	[0,1]
Alarm	10	[0,3,4,5,6,8,9,10,12,13]
Insurance	2	[0,5]
Water	6	[0,3,7,8,16,24]
Win95pts	10	[0,1,2,3,4,5,6,7,8,9]

TABLE 5 The experimental results of the learned model and its classification based on K2 algorithm

		The learned MBC model (Ave 5 times)			Classification based on the MBC (Ave 5 times)		
Name	**Methods**	***precision***	***recall***	***time(s)***	acc_G	acc_M	***time(s)***
Asia	K2	100.00±1.64	100.00±0.62	1.24±0.04	66.40±0.36	88.44±0.98	0.34±0.07
	Alg1+K2	72.73±0.58	100.00±0.58	2.38±0.07	65.77±0.33	88.39±0.80	0.37±0.96
	NB+**Alg2**	70.00±0.41	87.50±0.46	2.26±0.05	65.83±0.13	88.44±0.77	0.05±0.09
	Alg1+Alg2	78.72±0.56	92.38±0.82	2.51±0.04	**66.43**±0.30	**88.46**±0.53	0.37±0.07
Sachs	K2	100.00±1.51	100.00±1.09	15.96±0.03	67.60±2.01	81.84±1.88	0.63±0.05
	Alg1+K2	76.19±0.86	94.12±0.73	10.12±0.25	66.33±1.65	81.65±0.81	0.77±0.58
	NB+**Alg2**	78.57±1.14	64.71±0.93	3.65±0.03	55.43±0.45	75.93±0.59	0.08±0.14
	Alg1+Alg2	88.89±0.96	94.15±0.75	4.54±0.27	**67.70**±0.29	**82.38**±0.53	0.77±0.12
Alarm	K2	97.62±0.76	89.13±0.94	187.41±7.82	68.20±2.14	95.52±1.98	252.89±0.02
	Alg1+K2	72.41±2.43	91.30±1.38	132.35±5.14	63.30±2.20	95.24±0.46	331.14±0.87
	NB+**Alg2**	54.55±1.59	13.04±0.64	13.85±0.76	58.10±2.31	94.80±0.44	145.28±0.45
	Alg1+Alg2	70.70±0.62	79.39±0.76	28.84±0.84	**68.23**±1.65	**95.54**±0.75	282.38±0.13
Insurance	K2	100.00±1.15	82.69±0.54	120.52±5.56	**38.58**±1.23	67.50±2.46	1.87±1.25
	Alg1+K2	83.67±0.93	78.85±0.99	109.32±1.25	37.00±1.06	62.30±2.93	2.91±6.71
	NB+**Alg2**	50.00±1.19	9.62±0.46	4.91±1.23	35.07±1.54	61.13±0.79	0.12±3.68
	Alg1+Alg2	72.22±0.85	25.00±0.24	6.93±0.94	38.52±1.58	**67.51**±0.18	1.06±0.07
Water	K2	100.00±1.43	42.42±1.02	108.44±3.73	7.83±0.42	56.15±1.84	548.65±0.03
	Alg1+K2	78.38±0.84	43.94±0.88	84.48±9.69	5.25±0.35	50.04±1.40	405.63±0.15
	NB+**Alg2**	42.86±1.34	4.55±0.76	9.08±0.62	4.00±0.05	**57.65**±0.67	148.46±0.28
	Alg1+Alg2	62.75±0.94	19.31±0.54	15.78±0.45	**7.85**±0.21	56.12±0.85	370.57±0.13
Win95pts	K2	89.69±0.64	77.68±0.98	316.04±1.90	**78.84**±2.35	98.32±0.98	383.47±3.07
	Alg1+K2	57.53±0.75	75.00±0.99	318.46±3.68	76.80±2.06	97.28±2.08	450.04±4.35
	NB+**Alg2**	50.00±1.18	11.61±0.98	27.96±1.99	75.30±2.17	97.51±1.14	142.24±5.04
	Alg1+Alg2	64.49±0.67	61.60±0.75	51.24±1.63	78.81±0.74	**98.92**±0.32	346.11±0.35

In this experiment, we conduct five times on each sample data with the equal size and obtain their mean and standard deviation. In order to verify our proposed algorithm, we divide Algorithm 1 into two parts to illustrate our work. One part is learning bridge subgraph using Lines 4 to 17 of Algorithm 1 and this is also named **Alg1**. The other part is named **Alg2** that learns class subgraph and feature subgraph using Line 2 and from Lines 19 to 25 of Algorithm 1, respectively. In this experiment, the node orders are given according to DAG's topological structure of benchmark BNs. The experimental results by K2 algorithm and HC algorithm are shown in Tables 5 and 6, respectively.

In this subsection, we compare our proposed algorithm (**Alg1+Alg2**) with other three methods. The first one is K2 algorithm in Table 5 or HC algorithm in Table 6, which shows that we use K2 or HC algorithm to learn whole MBCs, the learned MBC models may be not CB-decomposable. The second one is **Alg1**+K2 in Table 5 or **Alg1**+HC in Table 6, which shows that the bridge subgraph is learned by **Alg1** and class subgraph and feature subgraph are learned by K2 or HC algorithm. The third one is NB+**Alg2**, which denotes that the bridge subgraph is learned by naive Bayesian method, feature subgraph is learned by Lines 19 to 25 of Algorithm 1, and class subgraph is learned by Line 2 of Algorithm 1 based on K2 or HC algorithm.

From Tables 5 and 6, we have that the following conclusions.

1. Our **Alg1+Alg2** algorithm has the highest accuracy acc_G and acc_M in most cases (18/24) that showed in bold. As well as it has the lowest the standard deviations of acc_G and acc_M in most cases (15/24). Moreover, the learning times of CB-decomposable MBC models decrease dramatically by comparing with the first two methods. The NB+**Alg2** has the shortest learning times in Table 5 and in Table 6, but it has worse classification accuracies, which shows that our algorithm not only has higher accuracies but also is more stable in most cases.

TABLE 6 The experimental results of the learned model and its classification based on HC algorithm

		The learned MBC model (Ave 5 times)			Classification based on the MBC (Ave 5 times)		
Name	**Methods**	***precision***	***recall***	***time(s)***	acc_G	acc_M	***time(s)***
Asia	HC	100.00±0.63	54.00±0.78	9.03±2.27	65.83±0.66	**88.44**±0.98	0.39±0.07
	Alg1+HC	66.67±0.83	85.71±0.98	4.07±0.07	65.60±1.36	88.32±0.80	0.34±0.96
	NB+**Alg2**	50.00±0.47	57.14±0.54	2.23±0.11	62.40±2.51	87.08±0.77	0.34±0.09
	Alg1+Alg2	52.63±0.13	62.50±0.42	2.34±0.03	**65.95**±0.50	88.25±0.53	0.33±0.07
Sachs	HC	0.00±0.00	0.00±0.00	177.87±5.61	44.77±3.24	70.15±1.88	0.59±0.05
	Alg1+HC	78.95±0.95	100.00±0.78	73.08±2.37	**66.60**±1.79	80.53±0.81	0.77±0.58
	NB+**Alg2**	52.94±0.75	50.25±0.89	5.72±0.38	63.63±0.37	80.80±0.59	0.67±0.14
	Alg1+Alg2	53.33±0.63	56.00±0.57	5.50±0.27	64.11±0.42	**81.99**±0.53	0.65±0.12
Alarm	HC	1.46±0.82	0.86±0.74	1067.85±8.82	45.83±3.5	92.46±1.98	215.56±0.02
	Alg1+HC	64.00±1.45	80.00±1.32	1865.89±57.16	60.90±2.20	94.33±0.46	311.35±0.07
	NB+**Alg2**	31.25±0.99	11.36±0.54	16.01±0.98	49.50±0.30	93.22±0.14	180.41±0.05
	Alg1+Alg2	42.47±0.72	29.38±0.79	26.21±1.16	**65.81**±0.16	**94.35**±0.65	200.48±0.03
Insurance	HC	11.11±0.63	0.95±0.54	7046.06±0.78	24.87±1.23	49.87±2.46	2.87±1.95
	Alg1+HC	82.98±0.84	78.00±0.99	52.97±106.62	35.73±1.96	61.27±2.93	2.05±6.91
	NB+**Alg2**	37.50±1.21	5.88±0.87	5.17±0.50	31.50±1.04	58.00±0.79	0.91±3.48
	Alg1+Alg2	45.59±0.85	12.06±0.45	6.02±0.70	**35.81**±1.05	**61.35**±0.18	0.95±0.07
Water	HC	89.47±0.99	28.81±1.12	1341.64±85.85	6.25±0.39	46.15±1.84	362.80±0.03
	Alg1+HC	78.26±0.96	31.03±0.78	779.12±32.31	6.10±0.42	49.17±2.10	367.82±0.15
	NB+**Alg2**	33.33±1.47	3.08±0.65	9.56±0.17	6.00±0.00	45.46±0.67	197.46±0.08
	Alg1+Alg2	45.71±0.76	10.00±0.38	12.81±0.24	**6.64**±0.11	**56.85**±0.15	236.81±0.03
Win95pts	HC	68.27±2.54	75.53±0.98	17847.40±51.95	**78.40**±3.25	97.61±0.98	408.98±3.07
	Alg1+HC	48.18±0.99	68.75±0.48	13856.35±10.73	73.50±2.46	97.00±2.48	460.78±7.25
	NB+**Alg2**	26.09±1.25	5.36±0.54	30.83±0.61	58.10±2.47	94.78±2.54	242.24±5.24
	Alg1+Alg2	27.68±0.98	15.04±0.47	49.24±1.48	75.32±0.64	**98.23**±0.12	334.49±3.65

2. Although the *precision* and *recall* of our learned CB-decomposable MBC model are lower than other algorithms in most cases, the classification accuracy is higher than other algorithms in most cases, which shows that by our algorithm, we can learn simpler CB-decomposable MBC models that have the same good classification performance. In addition, for HC algorithm in Table 6, the learned models depend on the initial network as well we set that the loop will end when the BIC scores do not improve any more in consecutive two times, so the learned CB-decomposable MBC models by HC algorithm in Table 6 may be worse than our algorithm sometimes.
3. For the inference times or classification times, it is related to the number of class variables. Alarm, Water, and Win95pts, three networks, have long inference times because of their bigger number of class variables that are shown in Table 4. The inference times based on the learned MBC model for classification are less than K2 algorithm or HC algorithm in most cases (8/12), the reason is that our algorithm is simpler and CB-decomposable. Especially the inference times decrease significantly when the parameters in MBC models are large.

To sum up, we can learn simper CB-decomposable MBC models with lower learning times and better classification performance in most cases than other BN classification methods.

4.5 | Comparing with other non-BN classification methods

In this subsection, we also compared with other three non-BN multidimensional classification methods, such as *k* neighbor, decision tree, and random forest. As well, we compare with some methods in Section 4.4, such as HC method, NB+**Alg2(HC)** method, **Alg1**+HC, **Alg1+Alg2(HC)**, and **Alg1+Alg2(K2)**. The average classification results of five times on six BNs dataset are shown in Figures 4 to 9. Note that acc_G is described

FIGURE 4 The average classification results on Asia BN

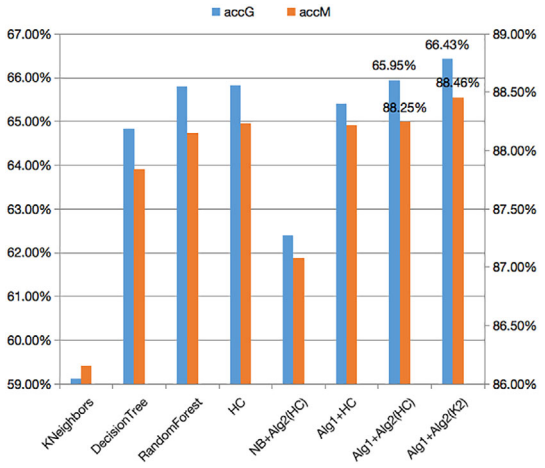


FIGURE 5 The average classification results on Sachs BN

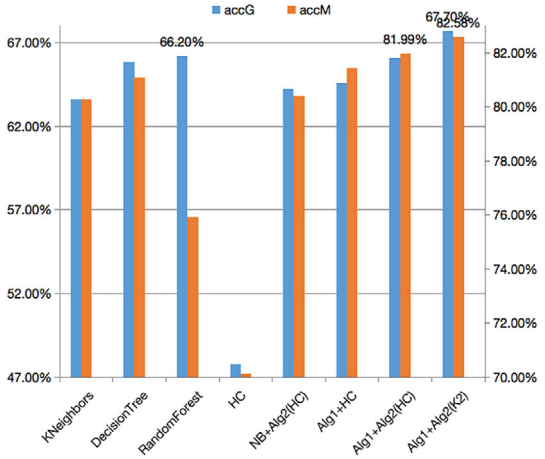
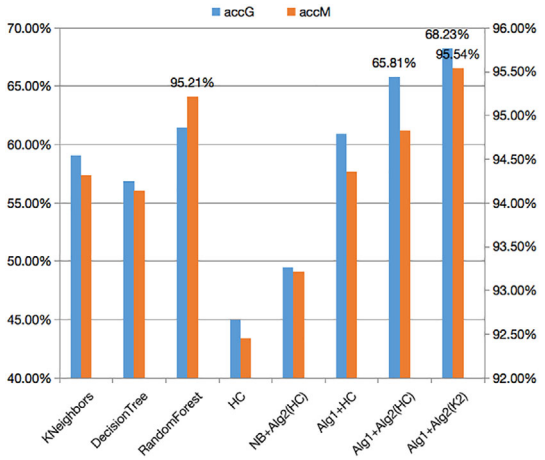


FIGURE 6 The average classification results on Alarm BN



on the left vertical axis in blue and acc_M is described on the right vertical axis in orange. The two vertical axis express different percentage ranges.

From Figures 4 to 9, we can have the following conclusions.

1. The classification performance of our Alg1+Alg2 algorithm, regardless of based on K2 or HC algorithm, is higher and more stable than other methods. In addition, Alg1+Alg2(K2) algorithm outperform Alg1+Alg2(HC) algorithm in acc_G and acc_M, because of being giving the node order in K2 algorithm. It is rather helpful for improving the model learning and its classification.

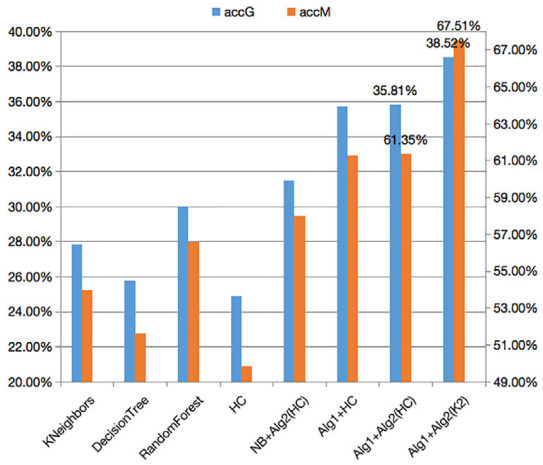


FIGURE 7 The average classification results on Insurance BN

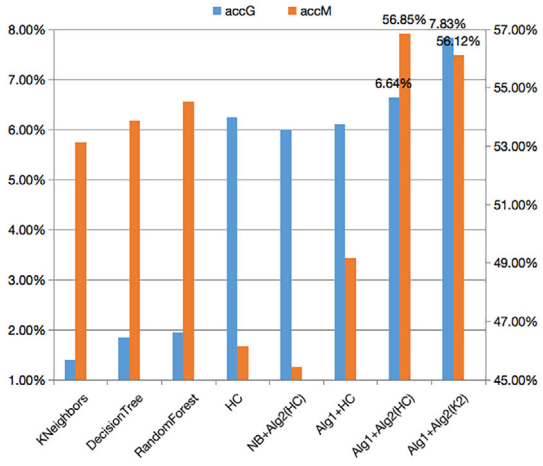


FIGURE 8 The average classification results on Water BN

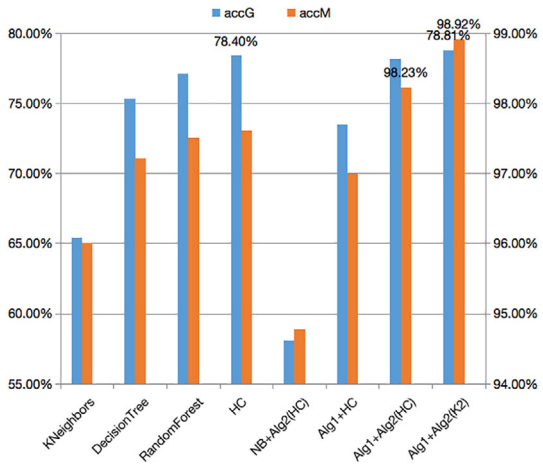


FIGURE 9 The average classification results on Win95pts BN

2. Similarly, for HC algorithm, it rather depends on its initial network. Moreover, the stoping condition of loop is that the BIC scores do not change in consecutive two times. Thus, HC algorithm is less stable than other methods in most cases.
3. For the two methods, NB+Alg2(HC) and Alg1+HC, the classification accuracies acc_G and acc_M based on Alg1+HC method are higher and more stable than those of based on NB+Alg2(HC) method. This illustrates that our Alg1 obtains more significant classification results than our Alg2 does.
4. The distances of $acc_M - acc_G$ show the fluctuation range between two highest points of acc_G histogram and acc_G histogram in each case. The range in our algorithm is smaller than any other methods. This demonstrates our algorithm can describe the relationships among class variables well, and it is easier to make the distances smaller than any other learning methods.

5 | CONCLUSION

In order to learn efficiently the structure of a CB-decomposable MBC that has tractability in the classification task, in this article, by studying the bridge subgraph learning strategies based on information gain ratio and updating feature subgraph strategies by bridge subgraph, we propose an efficient naive learning algorithm. Experimental results show the better performance of our proposed method, specifically it has not only higher accuracies, lower learning, and classification times but also simpler representation ability.

The proposed method also raises other issues. There are many missing data or noisy data in practice, how to learn the MBC models from missing data or noisy data will need us to study. Also, in this article, we only consider the discrete variables, in fact, the class variables that will be predicted include discrete as well as continuous variables. So we will study the multidimensional classification problems with the discrete and continuous variables. Besides, we also intend to do some applications of the MBC model, except for some ones that introduced in the Introduction, there are positioning or locating problems based on intelligent optimization algorithms[25-27] and preserving privacy problems.[28,29] To strengthen the classification performance in these applications, we will address multidimensional positioning and multidimensional preserving privacy in the corresponding scenarios, respectively. These are exactly our further work.

ACKNOWLEDGMENTS

This work was partially supported by the National Natural Science Foundation of China (Nos.61432011, U1435212, 61322211 and 61672332), the Postdoctoral Science Foundation of China (No.2016M591409), the Natural Science Foundation of Shanxi Province, China (Nos.201801D121115 and 2013011016-4), and Postgraduate Education Innovation Program of Shanxi Province (No. 2019SY352).

ORCID

Yali Lv https://orcid.org/0000-0001-5851-3931

REFERENCES

1. vander Gaag LC, de Waal PR. Multi-dimensional Bayesian network classifiers. Paper presented at: Proceedings of the 3rd European Conference on Probabilistic Graphical Models; 2006:107-114.
2. Zhou Y, Hospedales TM, Fenton N. When and where to transfer for Bayesian network parameter learning. *Expert Syst Appl*. 2016;55:361-373.
3. Zhao F, Huang Y, Wang L, Tan T. Deep semantic ranking based hashing for multi-label image retrieval. Paper presented at: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2015:1556-1564.
4. Zhang ML, Zhou ZH. Multi-label neural networks with applications to functional genomics and text categorization. *IEEE Trans Knowl Data Eng*. 2006;18(10):1338-1351.
5. Cui Z, Du L, Wang P, Cai X, Zhang W. Malicious code detection based on CNNs and multi-objective algorithm. *J Parall Distrib Comput*. 2019;129:50-58.
6. Cui Z, Xue F, Cai X, Cao Y, Wang G, Chen J. Detection of malicious code variants based on deep learning. *IEEE Trans Ind Inform*. 2018;14(7):3187-3196.
7. Zhang M, Zhang K. Multi-label learning by exploiting label dependency. Paper presented at: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD); 2010 Washington, DC.
8. Huang J, Li G, Huang Q, Wu X. Joint feature selection and classification for multilabel learning. *IEEE Trans Cybern*. 2018;48(3):876-889.
9. Zhang M, Zhou Z. A review on multi-label learning algorithms. *IEEE Trans Knowl Data Eng*. 2014;26(8):1819-1837.
10. Tsoumakas G, Katakis I. Multi-label classification: an overview. *Int J Data Warehous Min*. 2007;3(3):1-13.
11. Yang L, Wu XZ, Jiang Y, Zhou ZH. Multi-label learning with deep forest, national key laboratory for novel software technology; 2019. https://arxiv.org/abs/1911.06557.
12. Friedman N, Geiger D, Goldszmidt M. Bayesian network classifiers. *Mach Learn*. 1997;29:131-163.
13. de Waal PR, vander Gaag LC. Inference and learning in multi-dimensional Bayesian network classifiers. Paper presented at: Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning under Uncertainty, Lecture Notes in Artificial Intelligence no. 4724; 2007:501-511; Springer.
14. Qazi M, Fung G, Krishnan S, et al. Automated heart wall motion abnormality detection from ultrasound images using Bayesian networks. Paper presented at: Proceedings of the International Joint Conference on Artificial Intelligence; 2007:519-525.
15. Cai X, Wang P, Du L, Cui Z, Zhang W, Chen J. Multi-objective 3-dimensional DV-Hop localization algorithm with NSGA-II. *IEEE Sens J*. 2019;19(21):10003-10015. https://doi.org/10.1109/JSEN.2019.2927733.
16. Wang P, Huang J, Cui Z, Xie L, Chen J. A Gaussian error correction multi-objective positioning model with NSGA-II. *Concurr Comput Pract Exp*. 2019;32(5):e5464. https://doi.org/10.1002/cpe.5464.
17. Cui Z, Zhang J, Wu D, et al. Hybrid many-objective particle swarm optimization algorithm for green coal production problem. *Inf Sci*. 2020;518:256-271.
18. Cai X, Niu Y, Geng S, et al. An under-sampled software defect prediction method based on hybrid multi-objective cuckoo search. *Concurr Comput Pract Exp*. 2019;32(5):e5478. https://doi.org/10.1002/cpe.5478.
19. Benjumeda M, Bielza C, Larrañaga P. Tractability of most probable explanations in multi-dimensional Bayesian network classifiers. *Int J Approx Reason*. 2018;93:74-87.
20. Bielza C, Li G, Larrañaga P. Multi-dimensional classification with Bayesian networks. *Int J Approx Reason*. 2011;52(6):705-727.
21. Borchani H, Bielza C, Larrañaga P. Learning CB-decomposable multi-dimensional Bayesian network classifiers. Paper presented at: Proceedings of the 5th European Workshop on Probabilistic Graphical Models; 2010:25-32.
22. Cooper GF, Herskovits E. A Bayesian method for the induction of probabilistic networks from data. *Mach Learn*. 1992;9:309-347.
23. Quinlan J. *C4.5: Programs for Machine Learning*. San Mateo, CA: Morgan Kaufmann; 1993.

24. Jiang S, Zhang H. A fast decision tree learning algorithm. Paper presented at: Proceedings of the 21st National Conference on Artificial Intelligence AAAI'06, Vol. 1; (2006):500-505.
25. Cui Z, Zhang J, Wang Y, et al. A pigeon-inspired optimization algorithm for many-objective optimization problems. *Sci China Inf Sci*. 2019;62(7):70212.
26. Wang Y, Wang P, Zhang J, et al. A novel bat algorithm with multiple strategies coupling for numerical optimization. *Mathematics*. 2019;7(2):135.
27. Cui Z, Sun B, Wang G, Xue Y, Chen J. A novel oriented cuckoo search algorithm to improve DV-Hop performance for cyber-physical systems. *J Parall Distrib Comput*. 2017;103:42-52.
28. Hassan M, Rehmani M, Chen J. Differential privacy techniques for cyber physical systems: a survey. *IEEE Commun Surv Tutor*. 2020;22(1):746-789. https://doi.org/10.1109/COMST.2019.2944748.
29. Qi L, Zhang X, Dou W, Hu C, Yang C, Chen J. A two-stage locality-sensitive hashing based approach for privacy-preserving mobile service recommendation in cross-platform edge environment. *Future Generat Comput Syst*. 2018;88:636-643.
30. Rodríguez J, Lozano J. Multi-objective learning of multi-dimensional Bayesian classifiers. Paper presented at: Proceedings of the 8th International Conference on Hybrid Intelligent Systems; 2008:501-506.

How to cite this article: Lv Y, Hu W, Liang J, Qian Y, Miao J. A naive learning algorithm for class-bridge-decomposable multidimensional Bayesian network classifiers. *Concurrency Computat Pract Exper*. 2020;e5778. https://doi.org/10.1002/cpe.5778