Applied Intelligence
https://doi.org/10.1007/s10489-023-04999-2

A novel data enhancement approach to DAG learning with small data samples

Xiaoling Huang[1,2] · Xianjie Guo[1] · Yuling Li[1] · Kui Yu[1]

Accepted: 3 September 2023

Abstract
Learning a directed acyclic graph (DAG) from observational data plays a crucial role in causal inference and machine learning. However, the scarcity of observational data is a common phenomenon in real-world applications, where the current DAG learning methods may cause unsatisfactory performance in the context of small data samples. Data enhancement has been recognized as one of the key techniques for improving the generalization abilities of learning models utilizing small data samples. However, due to the inherent difficulty of sampling small datasets to generate high-quality new data samples, this approach has not been widely used in DAG learning. To alleviate this problem, we propose a data enhancement-based DAG learning (DE-DAG) approach. Specifically, DE-DAG first presents an integrated data sampling strategy for DAG learning and data sampling, then constructs a sample-level adaptive distance computing algorithm for selecting high-quality samples from the sampled datasets, and finally implements a DAG learning method on enhanced datasets consisting of high-quality samples and the original data samples. Experimental results obtained on benchmark datasets demonstrate that our proposed approach outperforms the state-of-the-art baselines.

Keywords Bootstrap sampling · Causal structure learning · DAG learning · Data enhancement

1 Introduction

Causal inference is one of the fundamental methods that is capable of producing economic value and promoting social development in the field of data science [1]. Learning a directed acyclic graph (DAG) from observational data is an important step for causal inference and robust machine learning methods. Specifically, a high-quality DAG is of critical significance for improving the effectiveness of these approaches.

✉ Xiaoling Huang
hxl@chzu.edu.cn

Xianjie Guo
xianjieguo@mail.hfut.edu.cn

Yuling Li
lyl95@mail.hfut.edu.cn

Kui Yu
yukui@hfut.edu.cn

[1] School of Computer Science and Information Engineering, Hefei University of Technology, 230601 Hefei, China

[2] School of Computer and Information Engineering, Chuzhou University, 239000 Chuzhou, China

In recent years, DAG learning methods have been extensively proposed, and they can be mainly divided into combinatorial-based optimization DAG learning [2, 3] and continuous optimization-based DAG learning [4, 5]. In light of the different learning strategies, the combinatorial optimization DAG learning methods can be subdivided into constraint-based DAG learning, score-based DAG learning and hybrid DAG learning approaches. Constraint-based methods make use of conditional independence tests to learn a DAG from observational data [6, 7], while score-based methods such as GES [8] and GGSL [9] learn the best DAG from observational data by taking advantage of score functions.

Moreover, hybrid DAG learning methods, such as MMHC [10] and SLL+C/SLL+G [11], combine the ideas of constraint-based and score-based methods. These fruitful methods consist of three steps: first learning the local skeleton of each variable from observational data, then constructing a global skeleton by splicing the local skeletons of all variables, and finally orienting the global skeleton by using independence tests [12, 13] or score functions [8, 14–16]. To solve the combinatorial-based constraint, continuous optimization DAG learning methods [4, 5, 17–20], such as

Published online: 15 September 2023

Notears and DAG-NoCurl, formulate the DAG learning problem as a continuous optimization problem, and learn a DAG from observational data by optimizing a weighted adjacency matrix.

Although the current DAG learning methods have made remarkable progress in terms of both accuracy and efficiency, they still achieve unsatisfactory DAG performance due to the inevitable quality problems of observational data [21, 22], such as small samples. In particular, most of existing DAG learning methods were designed for datasets with large numbers of data samples [10, 21–23] (e.g., 500 samples or more). When the number of data samples is insufficient (e.g., a small data sample is utilized), conditional independence (CI) testing or score computing becomes unreliable [24], which demonstrates the poor performance of the existing DAG learning methods. However, datasets with few data samples are a common phenomena in many real-world applications, creating an obstacle to accurate DAG learning.

In a scenario with small data samples, some missing key samples (or high-quality samples) will cause the results of the statistical tests between some variables (conditional independence tests) to contradict the real results. For example, the results of variable A and the variable B which were originally independent, may now be incorrectly as dependent. In contrast, if variable A and the variable B were originally dependent, they may now be incorrectly judged as independent. As a result, many additional erroneous edges are learned, and some correct edges are deleted.

According to the above discussions, a question naturally arises: can we generate new and key samples(or high-quality samples) to enhance the originally observational data, thereby yielding better performance using the DAG learning methods with the help of enhanced data?

Data enhancement [25–27] is devoted to generating more data, and it is widely applied in the fields of computer vision [28] and natural language processing. However, relatively few research findings have applied data enhancement techniques to DAG learning. This is attributed to the fact that DAG learning has strict restrictions and specifications. Furthermore, it is difficult to generate high-quality new data samples based on small data samples.

To address this issue, we propose a data enhancement approach, and our contributions are summarized as follows:

(1) We propose a novel DAG learning method with the capability of data enhancement capabilities, called DE-DAG, which can learn a high-quality DAG from small data samples. Specifically, DE-DAG first presents an integrated data sampling strategy to generate several subdatasets containing many new data samples, then designs a high-quality dataset construction strategy for selecting high-quality samples from the newly generated subdatasets, and finally performs a DAG learning method on the enhanced data containing both the originally observational data and the newly high-quality data.
(2) The main idea of DE-DAG is that it designs a high-quality dataset construction strategy. This strategy first develops a sample-level attention-based distance computing algorithm, called *adaptiveDis*, to determine whether a sample derived from a newly generated subdataset is close to the observational data in the distance space. Then, based on the distances between each sample and the observational data, this strategy selects high-quality samples from the newly generated subdatasets that are near the observational data in the distance space.
(3) Utilizing five benchmark BN datasets, we conduct extensive experiments to verify the effectiveness of DE-DAG, and the experimental results show that our proposed DE-DAG approach achieves better performance than the state-of-the-art DAG learning methods.

The remainder of the paper is organized as follows. The related work is briefly introduced in Section 2. Several basic notations, our proposed DE-DAG method and its corresponding algorithm are presented in Section 3. Extensive experiments used to evaluate the effectiveness of the proposed algorithm are shown in Section 4 and the paper is concluded in Section 5.

2 Related work

Causal inference is an important component of science and human intelligence, and causal structure learning is a prerequisite for causal inference [20, 29]. To discover causal structures from data, we typically make use of causal structure learning approaches [1], where a causal structure is generally defined for DAG (i.e., a Bayesian network).

Hence, we primarily focus on DAG learning approaches. In recent decades, many effective methods have been proposed for learning DAGs from observational data [1]. According to their learning strategies, these methods can be categorized as constraint-based [7], score-based, and hybrid-based methods. In addition, the hybrid-based methods are combinations of constraint-based methods and score-based methods.

Constraint-based DAG learning approaches, such as PC [6], PC-stable [30], MIIPC [2] and PC-CS PC [3], make use of conditional independence tests to learn the independence and correlation between variables derived from observational data, and construct a corresponding DAG based on the independence relationship between the variables. However, the accuracy of the DAGs learned by such methods depends on the quality of the input observational data. Specifically, when the data are insufficient, the DAGs learned by such methods

will have greatly deviate from the true structure, leading to low accuracy of the DAG learned.

In contrast, the score-based DAG learning methods, such as GES [8], GGSL [9] and MAHC [16], make use of a score function to search for a DAG that exhibits the highest degree of fitting with the observational data in all possible graph structure spaces. The challenge of score-based DAG learning lies in how to find a DAG with the highest score from the exponential graph structure search space. In an exhaustive search, every possible graph is considered and scored. Therefore, the utilized score functions and search strategies are the main factors that impact the effectiveness of score-based DAG learning methods.

To solve the combinatorial constraint, continuous optimization-based DAG learning methods have been proposed. For example, NOTEARS [4] learns a DAG from observational data by optimizing a weighted adjacency matrix, and formulates the acyclic constraint as a smooth term and solves the problem using gradient-based numerical methods. DAG-NoCurl [5] is proposed based on Hodge graph theory [31] to solve the resultant unconstrained optimization problem in the DAG space. Thus, the performance of the DAGs learned by these methods relies on not only the number of variables but also the quality of the observational data.

The abovementioned approaches improve the effectiveness of DAG learning to some degree, but they still face the scalability of the nodes in a DAG. Inspired by the combination of constraint-based and score-based methods, some hybrid DAG learning approaches, such as MMHC [10], SLL+G/C [11], PC+MCMC [32] and ADL [33] have been proposed to improve the performance of DAGs. Specifically, these methods first learn the local skeleton structure of each variable from the input observational data. Furthermore, the local skeleton structure of each variable is spliced into a global skeleton structure. Finally, the undirected edges in the global skeleton are oriented using a strategy built on constraint-based methods or score-based methods. Hence, a hybrid method not only avoids inaccurate orientation problem of constraint-based methods but also solves the problem of high time complexity of score-based methods. However, the impact of data quality on DAG learning methods has not been fundamentally addressed.

Although the existing DAG learning algorithms have achieved promising results, they rely heavily on the availability of sufficient observational data. However, DAG learning methods are developed under the assumption that "observation data is sufficient", this has led to relatively few research efforts related to DAG learning for small data samples. In practice, the volumes of observational data is extremely small in many cases. As a result, the existing DAG learning methods are incapable of achieving satisfactory performance with sparse observational data.

To overcome the data scarcity problem, data enhancement technology has been proposed in the field of computer vision [34] and natural language processing [35, 36]. These methods increase the amount of original data by adding slightly modified copies of the original data or synthetic data that are newly created from the existing data. Shorten et al. [34] investigated on the application of image-based data augmentation, which artificially increases the size of the training dataset through data distortion or oversampling. Li et al. [35] conducted a survey related to data enhancement approaches in natural language processing (NLP).

Although data enhancement is widely used in computer vision and natural language processing, it has received less attention in DAG learning scenarios. Different from images and natural language, it is more difficult to adopt data enhancement approaches in DAG learning. To alleviate scenarios with data scarcity in which DAG learning methods may fail, in this study, we introduce a data enhancement approach for generating abundant pseudo samples, which are jointly trained with the original observational data to increase both the quantity and the diversity of the observational data.

3 Proposed DE-DAG approach

In this section, we propose the DE-DAG approach which consists of three learning phases as shown in Fig. 1.

Phase 1: Generating new sampling datasets DE-DAG first samples datasets via the bootstrap method, and discovers the DAGs from each sampling dataset by using an existing DAG learning method. Furthermore, based on the sampling datasets, DE-DAG learns conditional probability tables for the learned DAGs using Bayesian network parameter learning methods. Finally, a new batch of datasets is generated based on the learned conditional probability tables and DAGs.

Phase 2: Selecting high-quality data samples Phase 2 first computes the distances between the data samples derived from the new batch of datasets obtained in Phase 1 and the original dataset, then selects some high-quality samples that match the original dataset, and finally places them into a high-quality dataset.

Phase 3: Relearning the DAG from the enhanced dataset In this phase, DE-DAG relearns a new DAG on the enhanced dataset consisting of the high-quality dataset and the original dataset.

Let $D = \{D_1^{m\times n}, \ldots, D_i^{m\times n}, \ldots, D_Q^{m\times n}\}$ denote a set of datasets named $sam\_datasets$, and $sam\_dataset\ D_i^{m\times n} = \{d_{i1}, \ldots, d_{ij}, \ldots, d_{im}\}$ denote the i-th dataset with n variables and m samples. $D_0^{m\times n} = \{d_{01}, \ldots, d_{0j}, \ldots, d_{0m}\}$ denotes the original dataset. Q represents the times of

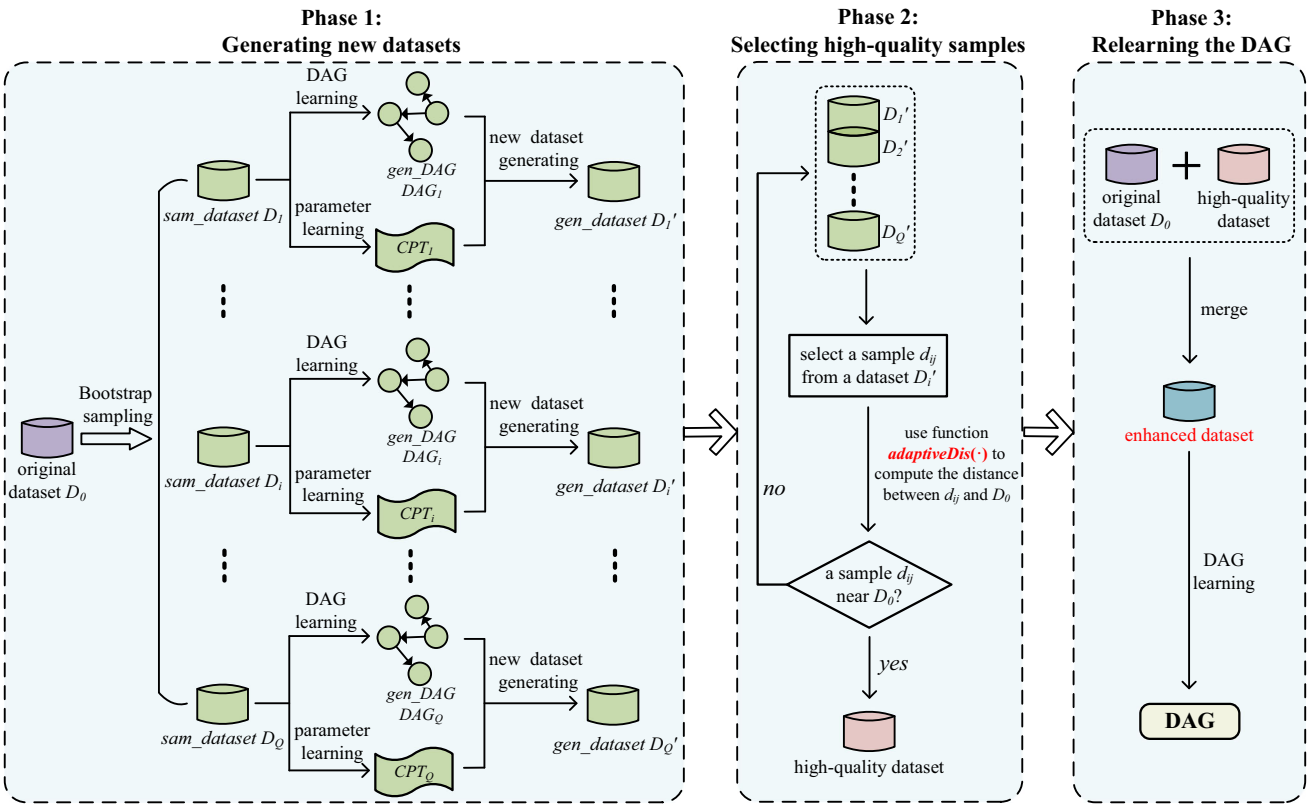


Fig. 1 The flow chart of our proposed DE-DAG approach

Bootstrap sampling. $DAG_i (1 \leq i \leq Q)$ named gen_DAG denotes the learned DAG based on a $sam\_dataset\ D_i^{m\times n}$. $CPT_i (1 \leq i \leq Q)$ means the parameter of conditional probability tables for $DAG_i (1 \leq i \leq Q)$. $D' = \{D_1'^{m\times n}, \ldots, D_i'^{m\times n}, \ldots, D_Q'^{m\times n}\}$ are named $gen\_datasets$, and $D_i'^{m\times n} = \{d_{i1}', \ldots, d_{ij}', \ldots, d_{im}'\}$ is a newly generated dataset based on a $gen\_DAG\ DAG_i$ and CPT_i.

In the following Sections 3.1 to 3.3, we focus on depicting the details of the aforementioned three phases.

3.1 Generating new sample datasets (Phase 1)

In Phase 1, we design an integrated data sampling strategy with the following Phases 1-1 and 1-2 for generating new data samples.

3.1.1 Learning an initial DAG for each sample dataset (Phase 1-1)

DE-DAG first samples new datasets from the original dataset by making use of a sampling method, and then learns the initial DAGs from these datasets.

Step 1 DE-DAG samples the original dataset $D_0^{m\times n}$ into Q datasets $D = \{D_1^{m\times n}, \ldots, D_i^{m\times n}, \ldots, D_Q^{m\times n}\}$ by using the bootstrap method [37].

The bootstrap method is a commonly used sampling method in the field of machine learning. The bootstrap idea of approximating a population by a sample becomes more credible as the sample size decreases, making it more suitable for even smaller datasets than other sampling methods. Because it can ensure that the subdatasets obtained through bootstrapping have the same dimensions as the original dataset, both in terms of rows and columns. However, other sampling methods may sample much smaller subdatasets than the original dataset, thus leading to unreliable statistical testing.

Furthermore, we select the bootstrap method for dataset sampling, because the bootstrap method is able to have 36.8% of its samples be different from those in the original dataset [37]. The sample difference can increase the diversity of the original dataset, which may be beneficial for learning more accurate DAGs from the sampled datasets than the DAGs derived from the original dataset. Hence, improving the diversity of the original dataset can help the DE-DAG (Phase 1 sampling_based method) to better generalize to unseen testing data.

Step 2: Based on each *sam_dataset* $D_i^{m\times n}$ obtained in Step 1, DE-DAG learns a DAG named *gen_DAG* $DAG_i (1 \leq i \leq Q)$ by using an existing DAG learning algorithm that is able to learn complete directed acyclic graphs without generating equivalent classes in a DAG.

3.1.2 Generating new sampling datasets (Phase 1-2)

Utilizing the DAGs learned in Phase 1-1, DE-DAG generates new datasets with the following two steps.

Step 1: Parameter learning DE-DAG learns the parameters of conditional probability tables for each DAG based on the *sam_dataset* and *gen_DAG* using the Bayesian network parameter learning method. Given a *sam_dataset* $D_i^{m\times n} (1 \leq i \leq Q)$, DE-DAG constructs a Bayesian network $bnet_i (1 \leq i \leq Q)$ based on *gen_DAG* $DAG_i (1 \leq i \leq Q)$. $BN = \{bnet_1, \ldots, bnet_i, \ldots, bnet_Q\}$ denotes a set of Bayesian networks containing both the learned DAGs and their conditional probability tables. Then, it learns the parameter of conditional probability table $CPT_i (1 \leq i \leq Q)$ by making use of $D_i^{m\times n}$ and $bnet_i (1 \leq i \leq Q)$

Step 2: Generating a new sampling dataset DE-DAG randomly generates new sampling datasets based on the BNs obtained in Step 1, which ensures that the newly generated samples conform to the data distribution of *samp-Datasets*. Specifically, DE-DAG constructs a new set of Bayesian networks named $BN' = \{bnet'_1, \ldots, bnet'_i, \ldots, bnet'_Q\}$, where $bnet'_i (1 \leq i \leq Q)$ consists of the parameter $CPT_i (1 \leq i \leq Q)$ and the corresponding structure information *gen_DAG* DAG_i. Furthermore, each new dataset named *gen_dataset* $D_i'^{m\times n} (1 \leq i \leq Q)$ is generated by the corresponding Bayesian network $bnet'_i (1 \leq i \leq Q)$. Accordingly, a set of *gen_datasets* $D' = \{D_1'^{m\times n}, \ldots, D_i'^{m\times n}, \ldots, D_Q'^{m\times n}\}$ is generated.

3.2 Selecting high-quality data samples (Phase 2)

Although we obtain a set of newly generated datasets in Phase 1, the quality of these datasets is different at each time due to the quality of the learned DAGs and their parameters. In addition, the newly generated samples increase the quantity of the original dataset, but there may exist some duplicate or noisy samples may be included, which may influence the performance of DAG learning. Therefore, it is necessary to select high-quality samples that match the original dataset from these newly generated datasets.

To select high-quality data samples, we first design an effective strategy for computing the distances between the original dataset and samples taken from the newly generated datasets. More detailed information is described in Section 3.2.1. Furthermore, according to these distances between the original dataset and the newly generated datasets, we select the samples from the newly generated datasets that are close to the original dataset in the distance space, and the detailed information is described in Section 3.2.2.

3.2.1 Computing the distances between the original dataset and the newly generated datasets (Phase 2-1)

Commonly-used distance metric methods cannot be directly employed to compute the distance between a sample point and a cloud of samples (i.e., the original dataset). Intuitively, this problem can be converted to computing the distance between the sample and the centre of the sample points acquired from the dataset. The conventional approach for computing the centre of sample points is to compute the average vectors of the sample points. Due to the sparsity of the data samples, the issue that one sample may be far from other samples can cause a massive deviation in the centre points of the dataset.

Inspired by the fruitful research [38] focusing more on the samples in relation to newly generated samples, we construct a sample-level adaptive distance computing algorithm (called *adaptiveDis*) to determine whether a sample obtained from the newly generated datasets belongs to the data distribution of the original dataset.

The detailed process of *adaptiveDis* is shown in Algorithm 1.

Algorithm 1 *adaptiveDis*

Require: Original dataset $D_0^{m\times n}$, *gen_datasets* $D' = \{D_1'^{m\times n}, \ldots, D_i'^{m\times n}, \ldots, D_Q'^{m\times n}\}$.
Ensure: The distance matrix *Distance* between D' and $D_0^{m\times n}$.
1: Initialization: $Distance = zeros(Q \times m, n+1)$. ▷ Construct a matrix containing $Q \times m$ rows and $n+1$ columns;
2: **for** $i = 1$ ***to*** Q **do**
3: $SamDistance = zeros(m, n+1)$.
4: **for** each sample $d'_{ij} (1 \leq j \leq m)$ in $D_i'^{m\times n}$ **do**
5: $SamDistance \leftarrow SamDistance \cup d'_{ij}$. ▷ Copy each sample d'_{ij} in $D_i'^{m\times n}$ to *SamDistance*;
6: **for** each sample $d_{0k} (1 \leq k \leq m)$ in $D_0^{m\times n}$ **do**
7: $weight_{jk} \leftarrow distance(d'_{ij}, d_{0k})$ based on Equation (1).
8: **end for**
9: **end for**
10: $CenterMatix_i^{m\times n} = Weight_i^{m\times m} \times D_0^{m\times n}$. ▷ Obtain an adaptive class centre for each sample in $D_i'^{m\times n}$;
11: **for** $j = 1$ ***to*** m **do**
12: $SamDistance(j, n+1) \leftarrow distance(d'_{ij}, CenterMatix_{ij})$ based on Equations (1) and (2).
13: **end for**
14: $Distance \leftarrow Distance \cup SamDistance$.
15: **end for**
16: **return** $Distance$.

Line 1 in Algorithm 1 Initializing the distance matrix. In Line 1, *adaptiveDis* constructs a distance matrix *Distance* containing $Q \times m$ rows and $n+1$ columns.

Lines 2 to 9 in Algorithm 1 Obtaining the weight of the adaptive centre of the original dataset for each sample of the newly generated datasets. In Lines 4 to 5 of Algorithm 1, the top n columns of each *SamDistance* are used to store samples from the i-th dataset $D_i'^{m\times n}(1 \le i \le Q)$. In Lines 6 to 8 of Algorithm 1, we argue that not all high-quality samples are always near the same centre of the original dataset. Hence, given the i-th dataset $D_i'^{m\times n}$, each sample $d'_{ij}(1 \le j \le m)$ in $D_i'^{m\times n}$ is set to a weight $weight_{jk}(1 \le j \le m, 1 \le k \le m)$ by computing the distance between the sample d'_{ij} and each sample $d_{0k}(1 \le k \le m)$ in the original dataset $D_0^{m\times n}$, which measures the deviation exhibited by the centre of the original dataset. All the samples weights are stored in the i-th matrix of $Weight_i^{m\times m}$.

To calculate the $distance(d'_{ij}, d_{0k})$ function, we employ the Euclidean distance measure, which is a commonly-used method for measuring spatial distance. Let $d'_{ij} = (s'_{j1}, s'_{j2}, \ldots, s'_{jn})$ and $d_{0k} = (s_{k1}, s_{k2}, \ldots, s_{kn})$ denote a sample from $D_i'^{m\times n}$ and a sample from the original dataset $D_0^{m\times n}$, respectively. Formally, the distance $distance(d'_{ij}, d_{0k})$ between d'_{ij} and d_{0k} is defined as follows.

$$distance(d'_{ij}, d_{0k}) = (|s'_{j1} - s_{k1}|^2 + |s'_{j2} - s_{k2}|^2 + \cdots + |s'_{jn} - s_{kn}|^2)^{\frac{1}{2}} \quad (1)$$

Line 10 in Algorithm 1 The adaptive centre of each sample in $gen\_dataset\ D_i'^{m\times n}$ is stored into $CenterMatix^{m\times n}$ by using the product of the normalized matrix $Weight_i^{m\times m}$ and the original dataset $D_0^{m\times n}$, where $Weight_i^{m\times m}$ is normalized between 0-1 to ensure the unification of its statistical probability distribution. Therefore, the element α_{0k} of the normalized matrix $Weight_i^{m\times m}$ can be expressed as follows:

$$\alpha_{0k} = \frac{exp(distance(d'_{ij}, d_{0k}))}{\sum_{p=1}^{m} exp(distance(d'_{ij}, d_{0p}))} \quad (2)$$

Lines 11 to 16 in Algorithm 1 Computing the distance between each sample and the adaptive centre of the original dataset. The $(n+1)-$th column of *SamDistance* is used to store the distance between each sample and the adaptive centre of the original dataset by using the Euclidean distance according to Equation (1). All the computed distances are stored in the *Distance* matrix.

3.2.2 Selecting the *K*-nearest neighbour samples from the newly generated datasets (Phase 2-2)

DE-DAG aims to select the nearest neighbour samples relative to the original dataset based on the computed distance *Distance*. *Distance* is sorted according to the $(n+1)-$th column of *Distance* in an ascending order, Then, we select the top-K samples from the sorted *Distance* matrix as high-quality samples.

3.3 Relearning the DAG from the enhanced dataset (Phase 3)

In this phase, DE-DAG first merges the high-quality samples obtained in Phase 2 into the original dataset to achieve data enhancement. In addition, it relearns a new DAG on the enhanced dataset consisting of the high-quality dataset and the original dataset by using an existing DAG learning method. In particular, we can employ any state-of-the-art DAG learning methods to learn the DAG without the constraint of generating a complete directed acyclic graph, as mentioned in Section 3.1.

4 Experiments

In this section, we design extensive experiments to demonstrate the effectiveness of the proposed DE-DAG approach.

4.1 Experimental settings

Comparison Methods. We compare the proposed approach with 9 state-of-the-art DAG learning methods, i.e., Peter-Clark (PC) [6], GES [8], MMHC [10], PC-stable [30], Notears [4], DAG-NoCurl [5], SLL+C/SLL+G [11] and GGSL [9].

Implementation details. PC, MMHC, PC-stable and our proposed DE-DAG approach are implemented in MATLAB. Furthermore, SLL+C/SLL+G and GGSL are implemented in C++, and the other approaches are implemented in Python. G^2 tests are utilized for the conditional independence tests with a statistical significance level of 0.01. The parameter Q of DE-DAG is set to 10. The random seed is set to -12 during the process of generating the original dataset, which ensures that generated data will be the same every time. With respect to Phase 1-2 of DE-DAG, we take advantage of the well-known Bayes Net Toolbox[1] named *BNT* to generate new sampling datasets. In addition, the other parameters in

[1] https://github.com/bayesnet/bnt

the compared algorithms are set as suggested in the corresponding literature. The source codes of PC, MMHC, and PC-stable are listed in the causal feature selection and structure learning package named *CausalLearner* [39], while the other source codes of GES, Notears and DAG-NoCurl are contained in the Causal Discovery Toolbox named *gCastle* [40].

Evaluation Metrics We adapt three metrics to evaluate the tested methods:

(1) Ar_F1. $Ar\_F1 = \frac{2 \times Ar\_Precision \times Ar\_Recall}{Ar\_Precision + Ar\_Recall}$. $Ar\_Precision$ represents the proportion of correctly directed edges in the learned graph among the edges output by an algorithm and Ar_Recall denotes the proportion of correctly directed edges in the learned graph out of the total edges in the true graph. The Ar_F1 score is the harmonic average of the $Ar\_Precision$ and Ar_Recall.
(2) Structural Hamming Distance (*SHD*): The *SHD* is the number of error edges including undirected edges, reverse edges, extra edges and missing edges.
(3) Running time (*Running time*): The *Running time* is the running time (in seconds) of each method.

In the following figures, (↑) means that higher values are better, and (↓) means that lower values are the better.

4.2 Benchmark datasets

We generate the original datasets according to five benchmark Bayesian networks (BNs) (as shown in Table 1) implemented in the *R* programming language by using the toolbox of *CausalLearner* toolbox. Each BN is used to generate 2 datasets with 50 and 100 samples, respectively, which is consistent with the scenario for small samples scenarios of DAG learning. Moreover, we choose the number of variables from 20 to 70 for the benchmark BNs.

Table 1 Summary of Benchmark BNs

Network	Num. Vars	Num. Edges	Data Size
Child	20	25	50/100
Insurance	27	52	50/100
Alarm	37	46	50/100
Haifinder	56	66	50/100
Hepar2	70	123	50/100

4.3 Comparison with the baselines

Figures 2, 3, 4 and 5 summarize the Ar_F1 and *SHD* values achieved with BNs utilizing 50 and 100 data samples. Specifically, DE-DAG first employs MMHC as the initial DAG learning method to learn a set of DAGs from divese sampled datasets, and then relearns the DAGs on an enhanced dataset using MMHC. In addition, we implement all the baselines on benchmark BNs with 50 samples and 100 samples, respectively. Taking the 50 samples case as an example, the results of DE-DAG are obtained on the enhanced dataset with 100 samples, which contains the 50 original data samples and 50 generated high-quality samples. Furthermore, when the volume of the benchmark BN dataset is 100, DE-DAG is performed on the enhanced dataset with 200 samples consisting of the 100 original data samples and 100 generated high-quality samples.

From Figs. 2-5, we can easily conclude that: 1) the performances of all the DAG learning methods drop dramatically when the volume of the original dataset volume is small, and 2) these methods achieve better results on the benchmark BN datasets with 100 samples than on those with 50 samples. It is proven that the volume of observational data volume is one of the important factors affecting DAG learning methods, and the results also indirectly indicates that our proposed DE-DAG approach based on data enhancement technology is of practical significance.

Table 2 shows the effectiveness and efficiency of all approaches on different benchmark datasets with 50 and 100 data samples. From Table 2, we can obtain the following observations.

(1) DE-DAG performs significantly better than the other methods with respect to the Ar_F1 metrics achieved on all datasets with 50 and 100 samples. Since DE-DAG achieves more balanced $Ar\_Precision$ and Ar_Recall values than its rivals, it is concluded that DE-DAG can learn more high-quality DAGs than the other methods. Furthermore, DE-DAG is better or at least has the similar *SHD* values to those of the best-performing baseline methods on all datasets. It is proven that DE-DAG is more suitable for original datasets with small samples than the other methods.
(2) The performance of DE-DAG does not depend on that of MMHC. Although MMHC has lower Ar_F1 values than the other methods on the datasets *Child*, *Insurance* and *Hailfinder* datasets with 50 samples, DE-DAG achieves the highest Ar_F1 values on all datasets. DE-DAG generates high-quality samples to enhance the original dataset, and thereby improves the accuracy of the learned DAG.

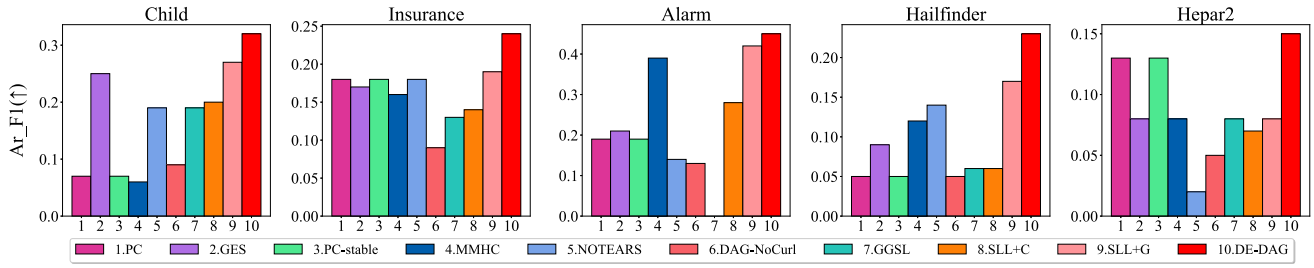


Fig. 2 Comparing results on benchmark BNs with 50 samples under Ar_F1

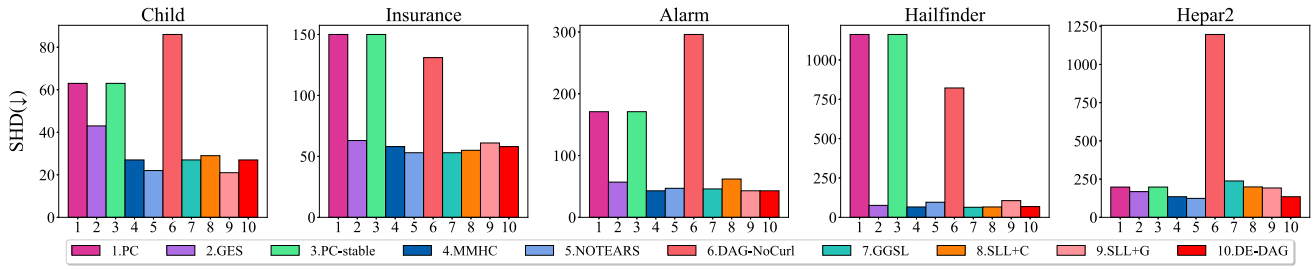


Fig. 3 Comparing results on benchmark BNs with 50 samples under *SHD*

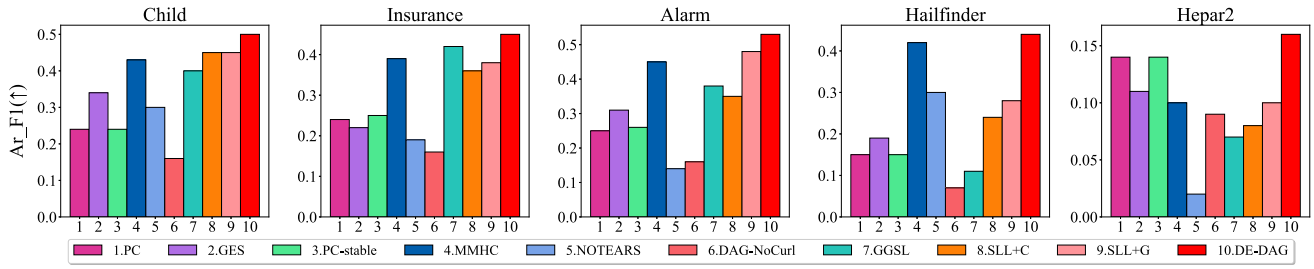


Fig. 4 Comparing results on benchmark BNs with 100 samples under Ar_F1

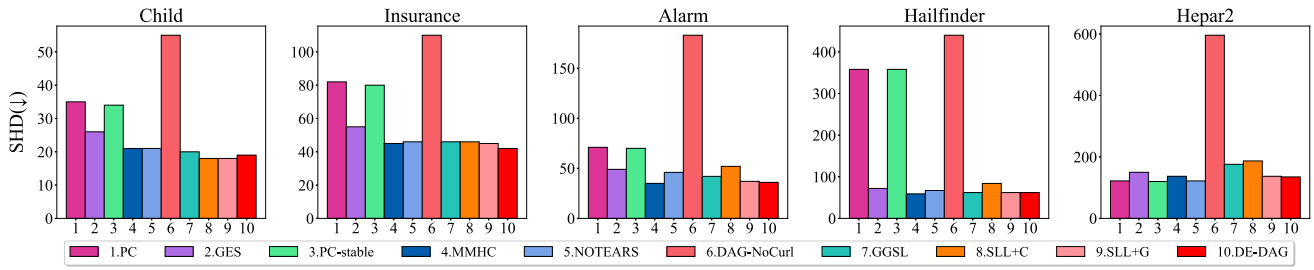


Fig. 5 Comparing results on benchmark BNs with 100 samples under *SHD*

Table 2 Comparing our proposed approach with different methods on 5 benchmark datasets when the size of original dataset is 50 and 100, repectively

Dataset	Method	#Samples	$Ar\_F1(\uparrow)$	$Ar\_Precision(\uparrow)$	$Ar\_Recall(\uparrow)$	$SHD(\downarrow)$	*Running time*$(\downarrow)$
Child	PC	50	0.07	0.05	0.12	63	0.31
	GES		0.25	0.20	**0.32**	43	15.91
	PC-stable		0.07	0.05	0.12	63	0.32
	MMHC		0.06	0.13	0.04	27	0.32
	Notears		0.19	**0.43**	0.12	22	0.97
	Dag-NoCurl		0.09	0.06	0.20	86	0.51
	GGSL		0.19	0.24	0.16	27	0.23
	SLL+C		0.20	0.40	0.13	29	**0.06**
	SLL+G		0.27	0.42	0.20	**21**	**0.06**
	DE-DAG		**0.32**	0.37	0.28	27	4.95
Insurance	PC	50	0.18	0.12	**0.35**	150	1.67
	GES		0.17	0.23	0.13	63	21.20
	PC-stable		0.18	0.12	0.35	150	1.58
	MMHC		0.16	0.25	0.12	58	0.41
	Notears		0.18	0.38	0.12	**53**	2.44
	Dag-NoCurl		0.09	0.06	0.13	131	2.15
	GGSL		0.13	**0.44**	0.08	53	0.50
	SLL+C		0.14	0.24	0.1	55	0.11
	SLL+G		0.19	0.28	0.14	61	**0.08**
	DE-DAG		**0.24**	0.33	0.19	58	4.65
Alarm	PC	50	0.19	0.12	0.48	171	2.91
	GES		0.21	0.23	0.20	57	25.51
	PC-stable		0.19	0.12	0.48	171	2.76
	MMHC		0.39	**0.48**	0.33	**43**	0.49
	Notears		0.14	0.36	0.09	47	1.90
	Dag-NoCurl		0.13	0.08	**0.50**	296	9.01
	GGSL		0	0	0	46	**0.01**
	SLL+C		0.28	0.31	0.26	62	1.20
	SLL+G		0.42	0.47	0.38	**43**	0.88
	DE-DAG		**0.45**	0.47	0.43	**43**	5.53
Hailfinder	PC	50	0.05	0.03	**0.45**	1162	1.79
	GES		0.09	0.15	0.06	76	46.34
	PC-stable		0.05	0.03	**0.45**	1162	1.77
	MMHC		0.12	0.31	0.08	66	0.96
	Notears		0.14	0.13	0.14	96	38.25
	Dag-NoCurl		0.05	0.02	0.30	822	31.86
	GGSL		0.06	1.00	0.03	**64**	**0.10**
	SLL+C		0.06	0.07	0.06	66	1.61
	SLL+G		0.17	0.18	0.16	106	1.48
	DE-DAG		**0.23**	**0.39**	0.17	68	7.80
Hepar2	PC	50	0.13	0.14	0.12	198	**0.82**
	GES		0.08	0.12	0.07	168	49.15
	PC-stable		0.13	0.14	0.12	198	0.95
	MMHC		0.08	0.19	0.05	135	0.97

Table 2 continued

Dataset	Method	#Samples	$Ar\_F1(\uparrow)$	$Ar\_Precision(\uparrow)$	$Ar\_Recall(\uparrow)$	$SHD(\downarrow)$	*Running time*$(\downarrow)$
	Notears		0.02	**0.25**	0.01	**124**	3.24
	Dag-NoCurl		0.05	0.03	**0.24**	1196	138.04
	GGSL		0.08	0.08	0.09	238	48.49
	SLL+C		0.07	0.08	0.07	199	4.99
	SLL+G		0.08	0.09	0.07	192	4.78
	DE-DAG		**0.15**	0.23	0.11	135	11.72
Child	PC	100	0.24	0.21	0.28	35	0.14
	GES		0.34	0.32	0.36	26	24.16
	PC-stable		0.24	0.21	0.28	34	0.13
	MMHC		0.43	0.53	0.36	21	0.38
	Notears		0.30	**0.63**	0.20	21	0.61
	Dag-NoCurl		0.16	0.12	0.24	55	1.18
	GGSL		0.4	0.42	0.38	20	0.20
	SLL+C		0.45	0.53	0.40	**18**	0.11
	SLL+G		0.45	0.53	0.40	**18**	**0.07**
	DE-DAG		**0.5**	0.58	**0.44**	19	4.41
Insurance	PC	100	0.24	0.21	0.29	82	0.20
	GES		0.22	0.29	0.17	55	32.50
	PC-stable		0.25	0.22	0.29	80	0.36
	MMHC		0.39	0.52	0.31	45	0.52
	Notears		0.19	**0.60**	0.12	46	2.26
	Dag-NoCurl		0.16	0.12	0.23	110	2.22
	GGSL		0.42	0.53	0.35	46	0.85
	SLL+C		0.36	0.48	0.29	46	0.17
	SLL+G		0.38	0.56	0.29	45	**0.14**
	DE-DAG		**0.45**	0.59	**0.37**	**42**	5.92
Alarm	PC	100	0.25	0.22	0.30	71	**0.23**
	GES		0.31	0.31	0.30	49	37.32
	PC-stable		0.26	0.22	0.30	70	0.27
	MMHC		0.45	0.50	0.41	**35**	0.63
	Notears		0.14	0.36	0.09	46	2.11
	Dag-NoCurl		0.16	0.10	0.39	183	5.96
	GGSL		0.38	0.42	0.35	42	2.51
	SLL+C		0.35	0.39	0.32	52	0.49
	SLL+G		0.48	0.47	0.49	37	0.43
	DE-DAG		**0.53**	**0.53**	**0.52**	36	7.30
Hailfinder	PC	100	0.15	0.09	**0.52**	358	1.46
	GES		0.19	0.32	0.14	72	52.77
	PC-stable		0.15	0.09	**0.52**	358	1.45
	MMHC		0.42	**0.53**	0.35	**59**	0.79
	Notears		0.30	0.37	0.26	67	39.43
	Dag-NoCurl		0.07	0.04	0.26	440	7.90
	GGSL		0.11	1.00	0.06	62	**0.17**
	SLL+C		0.24	0.26	0.23	84	4.23

Table 2 continued

Dataset	Method	#Samples	$Ar\_F1(\uparrow)$	$Ar\_Precision(\uparrow)$	$Ar\_Recall(\uparrow)$	$SHD(\downarrow)$	*Running time*$(\downarrow)$
	SLL+G		0.28	0.42	0.21	62	2.43
	DE-DAG		**0.44**	0.50	0.40	62	13.98
Hepar2	PC	100	0.14	0.43	0.08	122	0.32
	GES		0.11	0.18	0.08	150	57.74
	PC-stable		0.14	0.48	0.08	**120**	**0.28**
	MMHC		0.10	0.22	0.07	137	1.08
	Notears		0.02	**0.50**	0.01	122	2.97
	Dag-NoCurl		0.09	0.06	**0.25**	596	67.20
	GGSL		0.07	0.10	0.06	176	43.83
	SLL+C		0.08	0.09	0.07	187	2.84
	SLL+G		0.10	0.20	0.07	137	1.82
	DE-DAG		**0.16**	0.3	0.11	135	13.89

(3) The baselines perform poorly, due to the small data samples, leading to that these methods missing many true edges. SLL+C/SLL+G achieves a comparable performance to that of DE-DAG on *Child* with 100 samples due to its use of local learning methods. On most BNs, the performances of PC and PC-stable are close because they implement conditional independence tests, which require large numbers of data samples. Additionally, DAG-NoCurl has higher *SHD* values than the other methods on most datasets due to its strong theoretical assumptions.

(4) From the *Running time* metrics, we can easily see that although the proposed DE-DAG approach requires some calculations, it is not the slowest approach. GES runs slowly since it is a score-based algorithm, and its search space is relatively large. In general, the running times of all methods are not regular. For example, the running times of some methods under 50 samples are not necessarily shorter than those observed under 100 samples. Sometimes, hybrid DAG learning methods (e.g. SLL+G) have shorter running times than constraint-based DAG learning methods (e.g. PC), which may be caused by the small data samples environment.

To better understand the improvement provided by our proposed DE-DAG method, we employ DE-DAG as a basis to examine the *growth* and *speedup* metrics of all DAG learning approaches, where *growth* and *speedup* are defined by the following formulas, respectively.

$$growth = \frac{T_0 - T_1}{T_1} \tag{3}$$

where T_0 is the Ar_F1 value of our proposed DE-DAG method, and T_1 is the Ar_F1 value of the baseline DAG learning methods.

$$speedup(X) = \frac{X_1}{X_0} \tag{4}$$

where if X is the metrics of *SHD*, then X_0 denotes the *SHD* value of our proposed DE-DAG method, and X_1 denotes the *SHD* value of the baseline DAG learning methods. Hence, we name *speedup(SHD)* as *speedup(S)* for short. Similarly, if X is the metrics of *Running time*, then X_0 denotes the *Running time* of our proposed DE-DAG method, and X_1 is the *Running time* of baseline DAG learning methods. Hence, we name *speedup(Running time)* as *speedup(R)* for short.

Tables 3 and 4 further present changes in the *growth* and *speedup* metrics achieved on all datasets with 50 and 100 samples, respectively. DE-DAG increases the *growth* metrics by 7.14%-650.00% compared to those of the baseline approaches on all datasets with 50 samples, while the results of DE-DAG are increased by 4.76%-700.00% compared to those of baseline approaches on all datasets with 100 samples. For *speedup(S)*, the values vary from 0.78 to 17.09 compared to those of baseline approaches on all datasets with 50 samples, while the values change from 0.90 to 7.10 compared to those of baseline approaches on all datasets with 100 samples. From these evaluations, we can easily find that our proposed DE-DAG method outperforms the baselines in terms of Ar_F1 and achieves similar *SHD* values to those of the best-performing baseline.

For *speedup(R)*, the values vary from 0.01 to 11.78 compared to those of baseline approaches on all datasets with 50 samples, while the values change from 0.02 to 5.49 compared

Table 3 The outperformance of $growth$ and $speedup$ under our proposed DE-DAG against other approaches with 50 samples

Dataset	Method	#Samples	$growth(\uparrow)$	$speedup(S)(\uparrow)$	$speedup(R)(\uparrow)$
Child	PC	50	357.14%	**2.33**	0.06
	GES		28.00%	1.59	**3.21**
	PC-stable		357.14%	**2.33**	0.06
	MMHC		**433.33%**	1.00	0.06
	Notears		68.42%	0.81	0.20
	Dag-NoCurl		255.56%	3.19	0.10
	GGSL		68.42%	1.00	0.05
	SLL+C		60.00%	1.07	0.01
	SLL+G		18.52%	0.78	0.01
Insurance	PC	50	33.33%	**2.59**	0.36
	GES		41.18%	1.09	**4.56**
	PC-stable		33.33%	**2.59**	0.34
	MMHC		50.00%	1.00	0.09
	Notears		33.33%	0.91	0.52
	Dag-NoCurl		**166.67%**	2.26	0.46
	GGSL		84.62%	0.91	0.11
	SLL+C		71.43%	0.95	0.02
	SLL+G		26.32%	1.05	0.02
Alarm	PC	50	136.84%	3.98	0.53
	GES		114.29%	1.33	**4.61**
	PC-stable		136.84%	3.98	0.50
	MMHC		15.38%	1.00	0.09
	Notears		221.43%	1.09	0.34
	Dag-NoCurl		246.15%	**6.88**	1.63
	GGSL		**450.00%**	1.07	0.00
	SLL+C		60.71%	1.44	0.22
	SLL+G		7.14%	1.00	0.16
Hailfinder	PC	50	**360.00%**	**17.09**	0.23
	GES		155.56%	1.12	**5.94**
	PC-stable		**360.00%**	**17.09**	0.23
	MMHC		91.67%	0.97	0.12
	Notears		64.29%	1.41	4.90
	Dag-NoCurl		**360.00%**	12.09	4.08
	GGSL		283.33%	0.94	0.01
	SLL+C		283.33%	0.97	0.21
	SLL+G		35.29%	1.56	0.19
Hepar2	PC	50	15.38%	1.47	0.07
	GES		87.50%	1.24	4.19
	PC-stable		15.38%	1.47	0.08
	MMHC		87.50%	1.00	0.08
	Notears		**650.00%**	0.92	0.28
	Dag-NoCurl		200.00%	**8.86**	**11.78**
	GGSL		87.50%	1.76	4.14
	SLL+C		114.29%	1.47	0.43
	SLL+G		87.50%	1.42	0.41

Table 4 The outperformance of $growth$ and $speedup$ under our proposed DE-DAG against other approaches with 100 samples

Dataset	Method	#Samples	$growth(\uparrow)$	$speedup(S)(\uparrow)$	$speedup(R)(\uparrow)$
Child	PC	100	108.33%	1.84	0.03
	GES		47.06%	1.37	**5.48**
	PC-stable		108.33%	1.79	0.03
	MMHC		16.28%	1.11	0.09
	Notears		66.67%	1.11	0.14
	Dag-NoCurl		**212.50%**	**2.89**	0.27
	GGSL		25.00%	1.05	0.05
	SLL+C		11.11%	0.95	0.02
	SLL+G		11.11%	0.95	0.02
Insurance	PC	100	87.5%	1.95	0.03
	GES		104.55%	1.31	**5.49**
	PC-stable		80.00%	1.90	0.06
	MMHC		15.38%	1.07	0.09
	Notears		136.84%	1.10	0.38
	Dag-NoCurl		**181.25**%	**2.62**	0.38
	GGSL		7.14%	1.10	0.14
	SLL+C		25.00%	1.10	0.03
	SLL+G		18.42%	1.07	0.02
Alarm	PC	100	112.00%	1.97	0.03
	GES		70.97%	1.36	**5.11**
	PC-stable		103.85%	1.94	0.04
	MMHC		17.78%	0.97	0.09
	Notears		**278.57**%	1.28	0.29
	Dag-NoCurl		231.25%	**5.08**	0.82
	GGSL		39.47%	1.17	0.34
	SLL+C		51.43%	1.44	0.07
	SLL+G		10.42%	1.03	0.06
Hailfinder	PC	100	193.33%	5.77	0.10
	GES		131.58%	1.16	**3.77**
	PC-stable		193.33%	5.77	0.10
	MMHC		4.76%	0.95	0.79
	Notears		46.67%	1.08	2.82
	Dag-NoCurl		**528.57**%	**7.10**	0.57
	GGSL		300.00%	1.00	0.01
	SLL+C		83.33%	1.35	0.30
	SLL+G		57.14%	1.00	0.17
Hepar2	PC	100	14.29%	0.90	0.02
	GES		45.45%	1.11	4.16
	PC-stable		14.29%	0.89	0.02
	MMHC		60.00%	1.01	0.08
	Notears		**700.00**%	0.90	0.21
	Dag-NoCurl		77.78%	**4.41**	**4.84**
	GGSL		128.57%	1.30	3.16
	SLL+C		100.00%	1.39	0.20
	SLL+G		60.00%	1.01	0.13

to those of baseline approaches on all datasets with 100 samples, which indicates that the running time spent by the proposed DE-DAG method is of the same order of magnitude as those of the baseline methods. Furthermore, although the proposed DE-DAG approach requires some calculations, such as the distance matrix calculation, we can see that our DE-DAG is not the slowest approach. This is because the complexity of our proposed algorithm is $O(m^2)$, where m is the number of initial data samples. The problem studied in our work is aimed at small data samples, so the calculation cost of DE-DAG is not high.

To further compare the performance (in terms of Ar_F1) of DE-DAG with that of its rivals, we employ the Nemenyi test [41], which compares the difference between the average rankings of each pair of algorithms with a critical difference (CD) value. The CD for the Nemenyi test is defined as follows:

$$CD = q_{\alpha,r}\sqrt{\frac{r(r+1)}{6N}} \tag{5}$$

where α is the significance level, $|r|$ is the number of comparison approaches, N is the number of datasets. In our experiments, r=10, N=5, $q_{\alpha=0.05,r=10}$=3.164 at a significance level of α =0.05, and thus CD = 6.06.

Figure 6 provides the obtained CD diagrams, where the average rank of each approach is marked along the axis (lower ranks are shown to the right). We observe that DE-DAG is the only approach that achieves the lowest rank for different observational data. Specifically, when the size of the data samples increases, the rank value of DE-DAG is always 1.

4.4 Parameter sensitivity analysis

As mentioned in Section 3.2, we need to set the number of the original dataset's neighbours K in advance. To evaluate the influence of the parameter K when incorporating it into DE-DAG, we provide a sensitivity analysis of this parameter K in DE-DAG on different benchmark BN datasets.

To avoid the randomness of bootstrap sampling and verify our constructed *adaptiveDis* algorithm, DE-DAG is conducted on the same generated DAGs and datasets for a given benchmark dataset as the parameter K changes. Furthermore, we implement DE-DAG by varying the value of parameter K using the rates of 10%, 20%, 50%, 100%, 150%, 200%, 300%, and 400% of the size of the original dataset.

Figure 7 shows the sensitivity analysis conducted for the parameter K in DE-DAG on benchmark BNs. From the variational curves of Ar_F1 shown in Fig. 7(a) and (b), we can observe that DE-DAG achieves almost the best Ar_F1 value and achieves better *SHD* performance when the value of K is set to 50-75 and 100 on benchmark datasets with 50 and 100 samples respectively, which correspond to a rate of 100%-150% of the size of the original dataset.

From the variational *SHD* curves of depicted in Fig. 7(c) and (d), DE-DAG exhibits little change in the *SHD* metrics as the value of K increases. In particular, DE-DAG does not always have the lowest *SHD* when Ar_F1 is very high on benchmark datasets such as *Hailfinder* and *Hepar2*. The reason for this is that *SHD* only considers erroneous edges, while Ar_F1 considers not only erroneous edges, but also correct edges. DE-DAG achieves the highest Ar_F1 because it finds fewer missing edges, but obtains more extra edges than that of MMHC ($K = 0$) in the learned DAG.

Thus, the parameter K of DE-DAG is set to the rate 100% of the size of the original benchmark BN dataset in our experiments. Specifically, K is set to 50 when the size of the original benchmark BN has 50 samples, while K is set to 100 when the size of the original benchmark BN has 100 samples.

4.5 Effect of the adaptive distance computing algorithm

To demonstrate the effect of the high-quality samples selected by our proposed algorithm *adaptiveDis* algorithm, we select *Child* to show the distributions of the new generated samples and the original dataset by utilizing t-distributed stochastic neighbour embedding (t-SNE), which uses a probabilistic model to construct a mapping relationship between high-dimensional data points and low-dimensional embedding

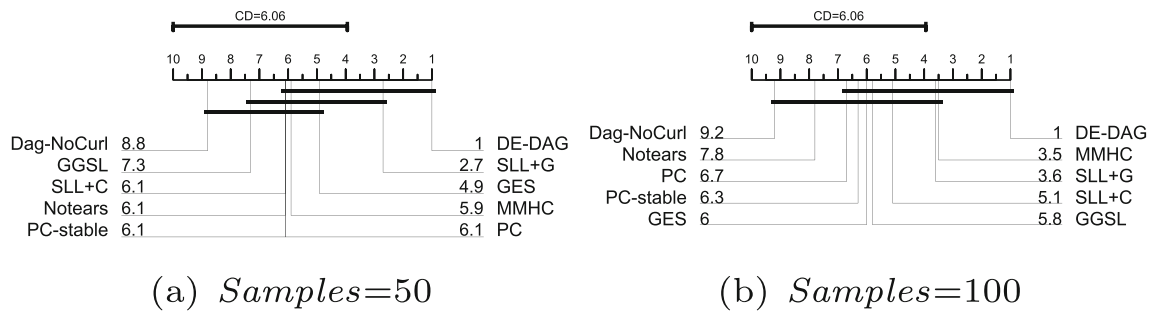


Fig. 6 Crucial difference diagram of the Nemenyi test for Ar_F1 on 5 benchmark BNs with different data samples

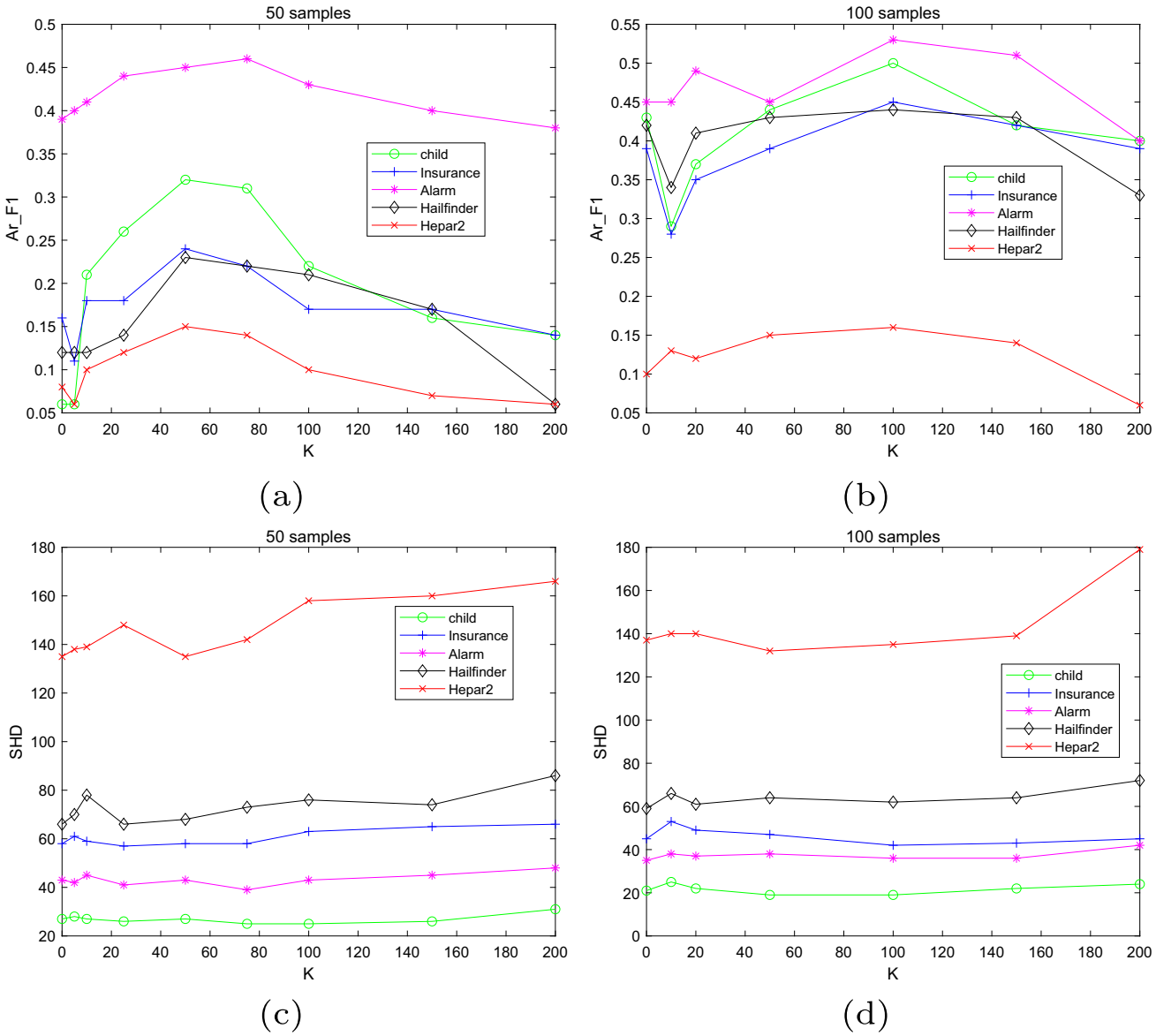


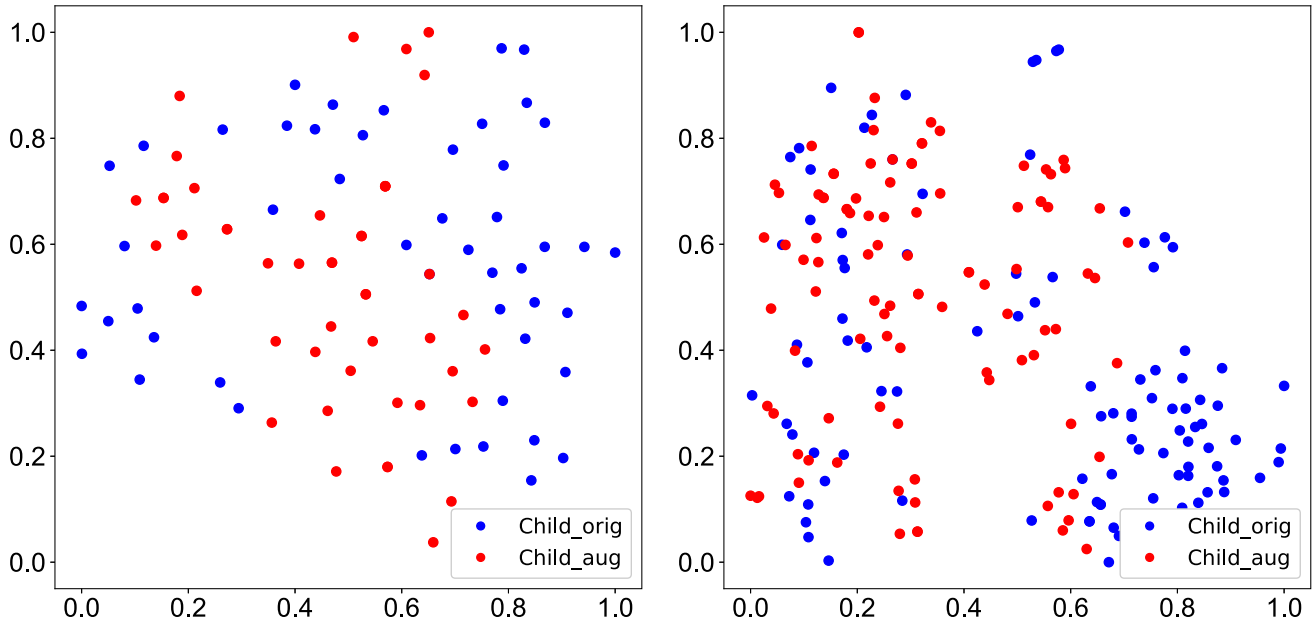
Fig. 7 *Ar_F*1 and *SHD* of DE-DAG on original datasets with 50 and 100 samples varying the value of K under benchmark BNs

points. In this subsection, we discuss the following two scenarios from a data distribution perspective.

Scenario 1: We employ *Child* to show the distributions of the high-quality samples produced by *adaptiveDis* and the original datasets from a graphic perspective. Figure 8 presents the distributions of the high-quality samples and the original datasets with 50 and 100 samples, respectively, where the samples from the original dataset are indicated as blue points, while the high-quality samples are indicated as red points. In Fig. 8, the x and y coordinates represent the coordinates of the data points in the reduced feature space, respectively. These coordinate values do not directly correspond to the original data feature values of the original data but rather represent a new feature representation obtained through a nonlinear mapping.

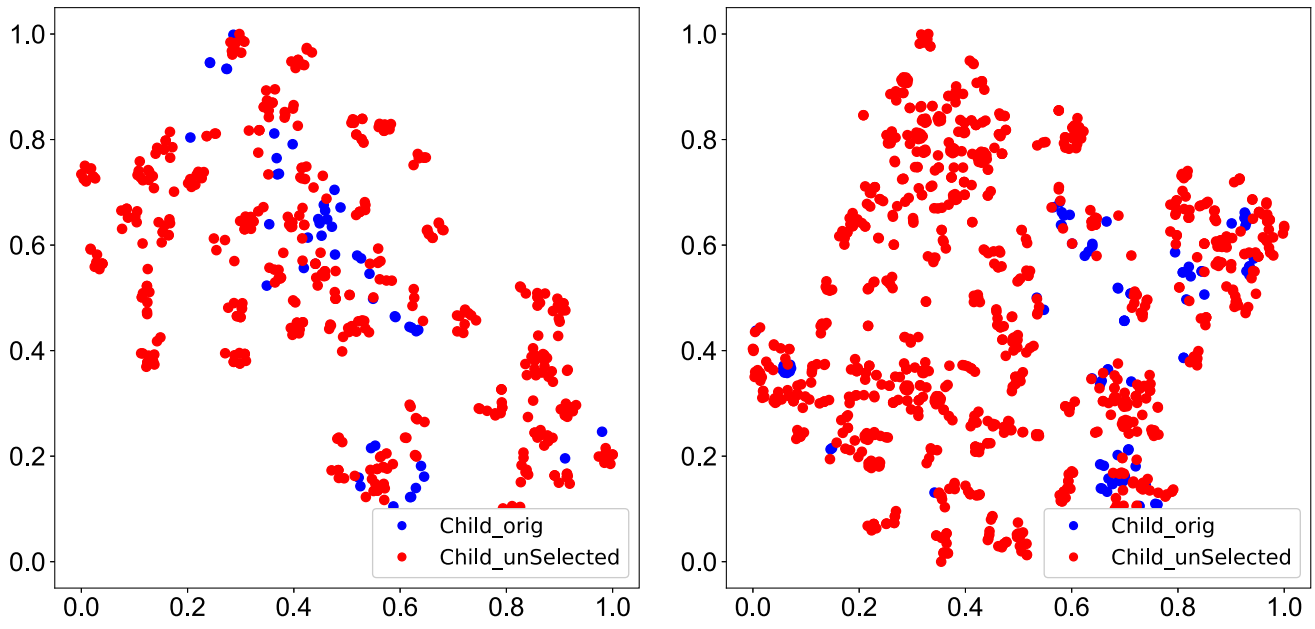
From Fig. 8, we can find that the high-quality samples often have three characteristics: 1) they increase the diversity of the original dataset, 2) the distribution of the new samples is close to that of the original dataset, and 3) the new samples can even fill the sparse areas in the distributions of the original datasets.

Scenario 2: We compare the distributions of the original datasets with those of the nonselected data samples, as shown in Fig. 9. In Fig. 9, we generate 500 samples on *Child* with 50 original data samples (indicated as blue points), selecting K=50 high-quality samples by using *adaptiveDis*, thus



(a) $Samples$=50, K=50 (b) $Samples$=100, K=100

Fig. 8 Comparing original data samples with high-quality samples



(a) $Samples$=50, non-K=450 (b) $Samples$=100, non-K=900

Fig. 9 Comparing original data samples with nonselected samples

leaving 450 nonselected samples (indicated as red points). Similarly, we generate 1000 samples on *Child* with 100 original data samples (indicated as blue points), selecting K=100 high-quality samples by using *adaptiveDis*, thus leaving 900 nonselected samples (indicated as red points).

From Fig. 9, we can observe that the nonselected data samples greatly overlap with the original data samples while deviating significantly from the original data samples.

Compared with Fig. 9, the high-quality samples in Fig. 8 are within the distribution of the original data, while the nonselected samples are outside the distribution of the original data. It is proven that the nearest neighbours of the adaptive centre selected for the original dataset can better select the high-quality samples, thereby improving the performance of DAG learning.

4.6 Rationale of the adaptive distance computing algorithm

In this subsection, we implement two other distance computing algorithms for DAG learning to demonstrate the rationale of the adaptive distance computing algorithm *adaptiveDis*.

Distance computing Algorithm 1 We construct a distance computing algorithm named $avgK$ to select high-quality samples. Specifically, we first compute the distances between each candidate sample from the new sampling datasets and each sample from the original dataset by using the Euclidean distance measure, then take the average of these distances for each candidate sample, and finally sort these candidate samples according to their average distances and select the top-K candidate samples as high-quality samples.

Distance computing Algorithm 2 We design another distance computing algorithm for high-quality sample selection, namely $minK$. With the second distance computing algorithm, we first calculate the distances between each candidate sample from the new sampling datasets and each sample from the original dataset by using the Euclidean distance measure. Then, we find the minimal distance for each candidate sample among these distances. Finally, we sort these candidate samples according to their minimal distances in descending order and select the top-K candidate samples as high-quality samples.

In our experiments, given the same new samples datasets generated in Phase 1, we conduct $avgK$, $minK$ and *adaptiveDis* to select high-quality samples, and combine them with the original dataset to relearn the DAGs. Table 5 depicts comparisons between the results of different distance computing algorithms and DE-DAG. We can see that our proposed *adaptiveDis* algorithm is significantly better than the other two algorithms on all the benchmark datasets, while the samples selected by the other two distance computing methods cannot effectively improve the performance of DAG learning well. This is because of the sparsity of the features in small data samples, which makes it difficult to find high-quality samples around the centre of the original data. Since *adaptiveDis* constructs a sample-level adaptive distance computing algorithm, it can better select high-quality samples for improving the effect of DAG learning with small data samples.

Table 5 Comparing our proposed approach with different distance computing algorithms on 5 benchmark datasets when the size of original dataset is 50 and 100, repectively

Dataset	Method	#Samples	$Ar\_F1(\uparrow)$	$SHD(\downarrow)$	#Samples	$Ar\_F1(\uparrow)$	$SHD(\downarrow)$
Child	DE-DAG-minK	50	0.17	34	100	0.29	27
	DE-DAG-avgK		0.14	31		0.33	22
	DE-DAG-*adaptiveDis*		**0.32**	**27**		**0.5**	**19**
Insurance	DE-DAG-minK	50	0.19	62	100	0.29	50
	DE-DAG-avgK		0.17	63		0.22	55
	DE-DAG-*adaptiveDis*		**0.24**	**58**		**0.45**	**42**
Alarm	DE-DAG-minK	50	0.44	41	100	0.44	39
	DE-DAG-avgK		0.43	44		0.48	37
	DE-DAG-*adaptiveDis*		**0.45**	**43**		**0.53**	**36**
Hailfinder	DE-DAG-minK	50	0.18	75	100	0.25	76
	DE-DAG-avgK		0.14	87		0.34	69
	DE-DAG-*adaptiveDis*		**0.23**	**68**		**0.44**	**62**
Hepar2	DE-DAG-minK	50	0.1	150	100	0.1	155
	DE-DAG-avgK		0.1	151		0.07	177
	DE-DAG-*adaptiveDis*		**0.15**	**135**		**0.16**	**135**

Case study To further validate the rationale of the adaptive distance computing algorithm of (*adaptiveDis*), we conducted a Kolmogorov-Smirnov test(KS test for short), which is a statistical method used to determine whether two datasets are drawn from the same underlying probability distribution. The KS test produces a p-value that quantifies the degree of similarity between the two distributions. The larger the p-value, the more similar the cumulative distribution functions of the two datasets are.

In our experiments, we use *Child* with 100 samples as the original dataset, then select 3 datasets composed of K=100 high-quality samples by using the $avgK$, $minK$ and *adaptiveDis* algorithms, and finally compare the distributions of these high-quality datasets with that of the same original dataset.

When we utilize the $minK$ algorithm for selecting high-quality samples, the p-value between the distribution of the high-quality samples and that of the original dataset is 6.45842e-15, while the p-value between the distribution of the high-quality samples and that of the original dataset is 4.98033e-15 when the $avgK$ algorithm is used for high-quality sample selection. When we use the *adaptiveDis* algorithm to select high-quality samples, the p-value between the distribution of the high-quality samples and that of the original dataset is 1.16899e-3. From these p-values, we find that although the distribution of high-quality samples selected by these algorithms are all inconsistent with that of the original dataset when the significance level α is set as 0.05, the distribution of high-quality samples selected by the *adaptiveDis* algorithm is the closest to that of the original dataset among those of the three algorithms. This further verifies that the samples selected by *adaptiveDis* possess higher-quality. Therefore, our proposed DE-DAG method selects high-quality samples by using *adaptiveDis*, and DE-DAG-*adaptiveDis* and DE-DAG refer to the same method.

5 Conclusions

In this paper, we propose a novel DE-DAG approach for performing DAG learning with small samples via data enhancement technology. DE-DAG first presents an integrated data sampling strategy for obtaining a set of newly generated datasets, then constructs a sample-level adaptive distance computing algorithm for selecting high-quality samples from those datasets to match the distribution of the original dataset, and finally learns a more accurate DAG using the enhanced dataset. Experimental results show that DE-DAG outperforms the baseline methods, and can be easily instantiated by any DAG learning algorithm that can produce complete directed acyclic graphs. Therefore, in the future, we intend to design a unified framework for improving the efficiency of existing DAG learning algorithms in cases with small data samples. In addition, DE-DAG achieves better performance than the existing DAG learning methods, but it cannot be applied to high-dimensional datasets. Our proposed distance computing algorithm is not sufficiently robust to adapt to datasets with very large networks($>$ 1000 nodes). Hence, we will consider designing a new high-quality sample selection method for performing DAG learning on very large networks.

Acknowledgements This work was supported in part by the National Key Research and Development Program of China under Grant 2021ZD0111801, by the National Natural Science Foundation of China under Grant 61876206, the Natural Science Foundation of Educational Commission of Anhui Province (No. 2022AH051099), the Excellent Young Talents Fund Program of Higher Education Institutions of Anhui Province (No. gxyq2022098) and by the Key Project of the Natural Science Foundation of Educational Commission of Anhui Province under Grants KJ2021A1065 and KJ2021A1064.

Author Contributions All authors contributed to the study conception and design. Xiaoling Huang: Conceptualization, Methodology, Software, Validation, Writing - original draft, Writing - review and editing. Xianjie Guo: Conceptualization, Methodology, Software, Writing - review and editing, Validation. Yuling Li: Investigation, Software, Visualization, Writing - review and editing. Kui Yu: Investigation, Writing - review and editing. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Availability of data and materials The datasets and code used during this study are available upon reasonable request to the authors.

Declarations

Ethics approval Not applicable.

Consent to participate Not applicable.

Consent for publication Not applicable.

Competing interests The authors declare that they have no competing interests.

References

1. Kitson NK, Constantinou AC, Guo Z, Liu Y, Chobtham K (2023) A survey of Bayesian Network structure learning. Artif Intell Rev 1–94
2. Qi X, Fan X, Wang H, Lin L, Gao Y (2021) Mutual-information-inspired heuristics for constraint-based causal structure learning. Inf Sci 560:152–167
3. Marella D, Vicard P (2022) Bayesian network structural learning from complex survey data: a resampling based approach. Stat Methods Appl 31(4):981–1013
4. Zheng X, Aragam B, Ravikumar P, Xing E (2018) Dags with no tears: Continuous optimization for structure learning. NeurIPS'18 31: 9492–9503
5. Yu Y, Gao T, Yin NY, Ji Q (2021) DAGs with no curl: An efficient DAG structure learning approach. ICML'21 139: 12156–12166

6. Spirtes P, Glymour C (1991) An algorithm for fast recovery of sparse causal graphs. Social Science Computer Review 9(1):62–72
7. Sadeghi K, Soo T (2022) Conditions and assumptions for constraint-based causal structure learning. Journal of Machine Learning Research 23(109):1–34
8. Chickering DM (2002) Optimal structure identification with greedy search. Journal of Machine Learning Research 3:507–554
9. Tian G, Fadnis KP, Campbell M (2017) Local-to-global Bayesian network structure learning. ICML'17 70: 1193–1202
10. Tsamardinos I, Brown LE, Aliferis CF (2006) The max-min hill-climbing Bayesian network structure learning algorithm. Machine Learning 65(1):31–78
11. Niinimaki, T., Parviainen, P (2012) Local structure discovery in Bayesian networks. UAI'12, 634–643
12. Zhang H, Zhang K, Zhou SG, Guan JH, Zhang J (2021) Testing independence between linear combinations for causal discovery. AAAI'21, 6538–6546
13. Zhang K, Peters J, Janzing D, Schölkopf B (2011) Kernel-based conditional independence test and application in causal discovery. UAI'11, 804–813
14. Huang BW, Zhang K, Lin YZ, Schölkopf B, Glymour C (2018) Generalized score functions for causal discovery. SIGKDD'18 1551–1560
15. Cussens J (2011) Bayesian network learning with cutting planes. UAI'11 153–160
16. Constantinou AC, Liu Y, Kitson NK, Chobtham K, Guo Z (2022) Effective and efficient structure learning with pruning and model averaging strategies. Int J Approx Reason 151:292–321
17. Ng, I., Ghassami, A., Zhang, K (2020) On the role of sparsity and dag constraints for learning linear dags. NeurIPS'20, 33: 17943–17954
18. Yu Y, Chen J, Gao T, Yu M (2019) DAG-GNN: DAG structure learning with graph neural networks. ICML'19 7154–7163
19. Zhang M, Jiang S, Cui Z, Garnett R, Chen Y (2019) D-vae: A variational autoencoder for directed acyclic graphs. NeurIPS'19 32: 1586–1598
20. Vowels MJ, Camgoz NC, Bowden R (2022) D'ya like dags? a survey on structure learning and causal discovery. ACM Comput Surv 55(4):1–36
21. Guo XJ, Wang YJ, Huang XL, Yang S, Yu K (2022) Bootstrap-based causal structure learning. CIKM'22 656–665
22. Yu K, Yang Y, Ding W (2022) Causal feature selection with missing data. ACM Transactions on Knowledge Discovery from Data 16(4):1–24
23. Cao YW, Yu K, Huang XL, Wang YJ (2022) A new skeleton-neural DAG learning approach. PAKDD'22 13280: 626–638
24. Guo X, Yu K, Cao F, Li P, Wang H (2022) Error-aware markov blanket learning for causal feature selection. Inf Sci 589:849–877
25. Dai X, Yuan X, Wei X (2022) Data augmentation for thermal infrared object detection with cascade pyramid generative adversarial network. Appl Intell 52:967–981
26. Dessureault JS, Massicotte D (2023) Explainable global error weighted on feature importance: The xGEWFI metric to evaluate the error of data imputation and data augmentation. Appl Intell. https://doi.org/10.1007/s10489-023-04661-x
27. Zhang C, Li X, Zhang Z, Cui J, Yang B (2023) BO-Aug: learning data augmentation policies via Bayesian optimization. Appl Intell 53:8978–8993
28. Zhang Y, Wang Q, Hu B (2023) MinimalGAN: diverse medical image synthesis for data augmentation using minimal training data. Appl Intell 53:3899–3916
29. Luo G, Zhao B, Du S (2019) Causal inference and Bayesian network structure learning from nominal data. Applied Intelligence 49:253–264
30. Colombo D, Maathuis MH (2014) Order-independent constraint-based causal structure learning. Journal of Machine Learning Research 15(1):3741–3782
31. Jiang XY, Lim LH, Yao Y, Ye YY (2011) Statistical ranking and combinatorial Hodge theory. Mathematical Programming 127(1):203–244
32. Kuipers J, Suter P, Mofa G (2022) Efficient sampling and structure learning of Bayesian networks. Journal of Computational and Graphical Statistics 31:639–650
33. Guo X, Yu K, Liu L, Li P, Li J (2023) Adaptive skeleton construction for accurate DAG Learning. IEEE Transactions on Knowledge and Data Engineering. https://doi.org/10.1109/TKDE.2023.3265015
34. Shorten C, Khoshgoftaar TM (2019) A survey on image data augmentation for deep learning. Journal of Big Data 6(1):1–48
35. Li B, Hou Y, Che W (2022) Data augmentation approaches in natural language processing: a survey. AI Open 3:71–90
36. Bayer M, Kaufhold MA, Reuter C (2022) A survey on data augmentation for text classification. ACM Computing Surveys 55(7):1-39
37. Efron B, Tibshirani RJ (1994) An introduction to the bootstrap. CRC Press, Boca Raton
38. Gao T, Han X, Liu Z, Sun M (2019) Hybrid attention-based prototypical networks for noisy few-shot relation classification. AAAI'19 33: 6407–6414
39. Ling Z, Yu K, Zhang Y, Liu L, Li J (2022) Causal learner: A toolbox for causal structure and markov blanket learning. Pattern Recognit Lett 163:92–95
40. Kalainathan D, Goudet O, Dutta R (2020) Causal Discovery Toolbox: Uncovering causal relationships in Python. J Mach Learn Res 21(37):1–5
41. Demiar J, Schuurmans D (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(1):1–30