Causal Discovery from Streaming Features

Kui Yu[1], Xindong Wu[1,2], Hao Wang[1], and Wei Ding[3]
[1]Department of Computer Science, Hefei University of Technology, Hefei, 230009, China
[2]Department of Computer Science, University of Vermont, Burlington, VT 05405, USA
[3]Department of Computer Science, University of Massachusetts-Boston, Boston, MA 02125, USA
ykui713@gmail.com; xwu@cs.uvm.edu; jsjxwangh@hfut.edu.cn; ding@cs.umb.edu

Abstract— **In this paper, we study a new research problem of causal discovery from streaming features. A unique characteristic of streaming features is that not all features can be available before learning begins. Feature generation and selection often have to be interleaved. Managing streaming features has been extensively studied in classification, but little attention has been paid to the problem of causal discovery from streaming features. To this end, we propose a novel algorithm to solve this challenging problem, denoted as CDFSF (Causal Discovery From Streaming Features) which consists of two phases: growing and shrinking. In the growing phase, CDFSF finds candidate parents or children for each feature seen so far, while in the shrinking phase the algorithm dynamically removes false positives from the current sets of candidate parents and children. In order to improve the efficiency of CDFSF, we present S-CDFSF, a faster version of CDFSF, using two symmetry theorems. Experimental results validate our algorithms in comparison with other state-of-art algorithms of causal discovery.**

Keywords - ***causal discovery; streaming features; Bayesian networks.***

I. INTRODUCTION

Traditional learning systems assume that all features are readily available from the beginning. This assumption, however, is often violated in real-world applications. For example, for the problem of texture-based image segmentation, a label is assigned to each pixel in a training image according to its texture type. Texture is a property of a pixel's neighborhood, so we have a large number of different "texture filters" that can be applied to each neighborhood in the training image to generate features for the pixel. A training image for this task might easily contain tens of thousands of labeled pixels, and each filter might be costly to apply [10,16]. Not much attention on feature selection has been given to the reality that these potential features might be very expensive to generate and store.

An intriguing question from this case is that when we need a lot of computational efforts to generate those features up front, should we develop a new way to integrate the new features as they arrive and begin the computation, or spend a long time waiting for all features to be generated and then adopt an existing learning algorithm? The answer to this question is very important, since the large computational efforts to generate features provide a new learning situation where not all features can be generated up front and this solicits for new effective learning methods. To address this issue, the concept of streaming features was proposed [16,22-23]. With streaming features, we don't need to generate all features before learning begins. Instead, feature generation and selection are interleaved while the number of observations is left constant. Managing streaming features has received much attention in the research area of classification [10,12,16,22-23].

Meanwhile, there is little attention paid to causal discovery in the context of streaming features. Causal discovery is of fundamental and practical interest in many areas of science and technology, including biology, medicine and pharmacology. Causal discovery is applicable not only to causal structure learning, but also to feature selection for classification [1,3,4,5]. Since existing algorithms for causal discovery commonly assume that all features are given in advance, mining causal relations from streaming features is still an unexplored area.

To this end, we attempt to address this challenging problem and present a novel CDFSF (Causal Discovery From Streaming Features) algorithm in this paper. In order to further improve its efficiency, we use two symmetry theorems to present an S-CDFSF (Symmetrical CDFSF) algorithm, a faster version of the CDFSF algorithm.

The rest of the paper is organized as follows. The background is given in Section 2. Our algorithms are presented in Section 3. Experimental results are provided in Section 4. Section 5 concludes the paper with some discussions on future work.

II. BACKGROUND

Causal modeling and discovery are central to science and technology. Learning Bayesian networks is one of the most common methods to explore causal relations from the observed data.

Previous research efforts on causal discovery need to first learn a full Bayesian network to uncover causal relations among features [6-9,19]. Because of the advent of massive datasets in biology, finance, the WWW, and so on, datasets often involve hundreds of thousands of features, which has posed a serious challenge to existing algorithms of Bayesian network learning. Discovery of local causal relations without learning a full Bayesian network in advance has received considerable attention during the last few years in machine learning [1,4,5,11,17-18,21]. In general, local causal discovery as an effective means when dealing with hundreds of thousands of features is emphasized on two specific tasks:

(a) identification of features that are direct causes or direct effects of a target, and (b) discovery of Markov blankets.

In a faithful Bayesian network, the first task is to find the parents and children of a target of interest (PC(T) for short). Two major algorithms HITON_PC and MMPC for the discovery of PC(T) were introduced by Aliferis and Tsamardinos [3,4]. The second task, discovery of the Markov blanket of a target (MB(T) for short), is to find the set of parents, children, and parents of children for a target of interest. Representative work includes the GS, IAMB HITON_MB, MMMB, and PCMB algorithms [1,3,4,14,18].

Those existing algorithms above have significantly advanced the techniques for causal discovery in traditional learning environments where all features are given in advance. In the context of streaming features, the features are not presented up front any more. Thus, for mining causal relations from streaming features, the major challenges are twofold.

One is the increasing feature volumes. Different from traditional causal learning where all features are presented initially, in the context of streaming features, the features are generated dynamically and arrive one at a time. Aggregating all features to discover causal relations is practically infeasible for streaming features with continuous volumes. One has to devise new causal learning methods for streaming features.

The other is the ongoing discovery of causal relations for all arrived features. Existing causal learning algorithms can find causal relations for a target of interest at each run. In the context of streaming features, when the process of generating features is over, the causal relations for all features generated so far, are required to be achieved. Thus, mining causal relations must be done in the course of the process of generating features. One has to provide effectively ongoing mechanisms to integrate the new features as they arrive and begin the computation immediately.

Therefore, in this paper, we provide a study on causal discovery with streaming features and present two novel algorithms, CDFSF and S-CDFSF, to explore those major challenges.

CDFSF is presented with two phases: growing and shrinking. In the growing phase, CDFSF finds the set candidate parents and children of a target T (CPC(T) for short) for each feature seen so far. In this phase, when a new feature arrives, CDFSF takes each feature seen so far as a target T and assesses whether the new one belongs to its CPC(T). And then it regards the new one as a target T and discovers its CPC(T) from the features arrived so far. The shrinking phase dynamically removes the false positives from CPC(T) of the features seen so far. When the process of generating features is over, the sets CPC(T) for all of the features arrived so far, not for a special one of interest, are achieved. In order to improve the efficiency of CDFSF, we plug in the symmetry check, and present an S-CDFSF algorithm.

Both of our algorithms focus on the task of discovery of PC(T). Since the existing algorithms for the discovery of MB(T) consists of two steps: identification of PC(T), and then discovery of the candidate spouses of the target from the candidate parents of each feature within PC(T). The key difference between our algorithms and the existing algorithms is of identification of PC(T).

In this paper, a Bayesian network is used as the language to represent data generating processes and causal relations. In the rest of the paper, we will use the terms "variable", "node", and "feature" interchangeably.

III. PROPOSED ALGORITHMS

A. The CDFSF algorighm

In this section, we develop a novel algorithm, called CDFSF, to address the challenges on causal discovery with streaming features. One key work of CDFSF is how to efficiently discover causal relations between a new arriving feature X and a target feature T.

Definition 1 (Conditional independence) [15, 20] In a variable set V, two variables X and Y are conditionally independent given a set of variables Z, iff

$P(X \mid Y, Z) = P(X \mid Z)$, denoted as Ind(X,Y|Z).

For notational convenience we will denote conditional dependence as Dep(X,Y|Z).

Theorem 1 [15] In a faithful Bayesian network, there is an edge between the pair of nodes X∈ V and Y∈ V iff

$$\forall S \subseteq V \setminus \{X, Y\}, s.t.\ Dep(X, Y \mid S)$$

From Theorem 1, when a new feature X arrives, we can get an immediate method to discover causal relations between X and a target T: for any feature X∈ V\{T}, find a subset Z⊆V\{X,T} and test whether Ind(X,T|Z). If such a subset Z exists such that Ind(X,T|Z), then X∉ PC(T), otherwise X∈ PC(T). The major drawback of this method is that the set V is not given before learning in the context of streaming features. The method is also very inefficient since it needs to test all subsets of the set V excluding X and T even if the set V is known in advance. In terms of improving the efficiency, Aliferis et al. proved the following Corollary 1 which shows that to determine whether Ind(X,T|Z), it is not necessary to test all subsets Z⊆V\{X,T}, but all subsets Z⊆PC(X) \T and all Z⊆PC(T) \X.

Corollary 1 [4] In a faithful Bayesian network, there is an edge between the pair of nodes X∈ V and Y∈ V iff Dep(X,Y|S), for all S⊆PC (X)\{Y} and S⊆PC (Y)\{X}.

Clearly, if we knew the sets true PC(X) and PC(T) in a Bayesian network, then the number of subsets that would need to be checked for each X∈ PC(T) could be significantly reduced. But the problem is that in a real-world problem, we cannot get the sets true PC(X) and PC(T). Thus, [4] provides the following Corollary 2 which shows that we can work on a superset of true PC(X) or PC(T) .

Corollary 2 [4] In a faithful Bayesian network, there is an edge between the pair of nodes X∈ V and Y∈ V iff Dep(X,Y|Z), for all Z⊆S where PC(X) \Y⊆S⊆V\{X,Y}.

According to Corollary 2, if not all features in the set V are available in advance, we can work on the current CPC(T) of a target T to efficiently determine whether a new arriving feature X belongs to CPC(T) or not.

The other key contribution of the CDFSF algorithm is how to provide effectively ongoing mechanisms to discover causal relations among all generated features with increasing feature volumes.

CDFSF adopts an online mechanism for this task. This mechanism consists of two phases: the growing phase and shrinking phase. In the growing phase, when a new feature X arrives, CDFSF firstly takes each feature seen so far as a target T, and assesses whether X belongs to its CPC(T). Next, CDFSF regards X as the target, and then discovers its CPC(X) from the features arrived so far in turn. In the shrinking phase, CDFSF reevaluates each feature within the sets of candidate parents and children for all arrived features and eliminates false positives from them. When the process of generating features is over, the sets CPC for all features generated so far are returned.

The pseudo-code of CDFSF is shown in Table I. In Table I, the set GFS stores the features arrived so far, $CPC(X_i)$ denotes the set of candidate parents and children of X_i, and conditional independence tests use G^2 test which is a modified version of chi-square.

TABLE I. THE PSEUDO-CODE OF CDFSF

Input: streaming features X; **Output:** CPC(X),X={X_1,X_2,…,X_n}
1. Initialization
GFS={},CPC(X)={};
2. The growing phase
(1) Generate a new feature X_i
(2) GFS=GFS $\cup$ X_i
(3) For each feature T$\in$ GFS\ X_i , according to Corollary 2
if $\forall S \subseteq CPC(T)$ s.t Dep(X_i, T | S)
CPC(T)= CPC(T) $\cup$ X_i.
if $\forall Z \subseteq CPC(X_i)$ s.t Dep(T, X_i | Z)
CPC(X_i)= CPC(X_i) $\cup$ T.
3.The shrinking phase
For each feature Y $\in$ CPC(T), T$\in$ GFS, according to Corollary 2
if $\exists S \subseteq CPC(T) \setminus Y$ s.t Ind(Y, T | S)
CPC(T)= CPC(T)-Y.
4. Interleave step 2 and step 3 until all features arrive.

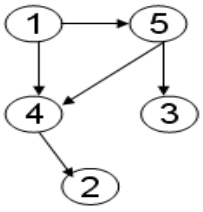

Figure 1. An illustrating Bayesian network to demonstrate the trace of the CDFSF algorithm

For instance, a trace of the algorithm is provided for data sampling from the example Bayesian network in Fig. 1. We assume that the network is faithful and so the conditional dependencies and independences can be read off the graph directly using the d-separation criterion. Now we assume that the set V={1,2,3,4,5} and the set ordering={5,1,4,2,3} which denotes the ordering of arrived features. We initialize GFS={} and CPC(X)={}. A trace of CDFSF is as follows.

(1) First, feature 5 arrives. The set GFS is an empty set excluding feature 5 so that CPC(5)={}.

(2) Feature 1 arrives where GFS={5,1}, CPC(5)={} and CPC(1)={}. In the growing phase, feature 5 inside GFS first is regarded as a target, and then our approach determines whether 1$\in$ CPC(5). Since there is all subsets of CPC(5) that makes 1 conditionally dependent of 5: Dep(1,5|ϕ), we get 1$\in$ CPC(5). Next, feature 1 is regarded as a target and our method tests whether 5 $\in$ CPC(1). In a similar way, 1$\in$ CPC(5). Since there are new features being added into CPC(5) and CPC(1), the shrinking phase is performed. Finally, CPC(5)={1}, and CPC(1)={5}.

(3) Feature 4 arrives where GFS={5,1,4}, CPC(5)={1} and CPC(1)={5}. In the growing phase, feature 5 inside GFS is first regarded as a target, then our method tests whether the new feature 4$\in$ CPC(5). Since all subsets of CPC(5) that makes 4 conditionally dependent of 5: Dep(4,5| ϕ) and Dep(4,5|1), feature 4 enters CPC(5), and then CPC(5)={1,4}. Then the shrinking phase reevaluates each feature within CPC(5) to remove false positives. Our method regards feature 1 as a target. As for testing whether feature 4 $\in$ CPC(1), in a similar way, we can get 4 $\in$ CPC(1). Finally, our method keeps the new arriving feature 4 as a target, and discovers CPC(4) from the set {5,1}. When the phase is over, CPC(5)={1,4}, CPC(1)={5,4} and CPC(4)={5,1}.

In a similar way after the arrival of features 2 and 3,the final outputs are CPC(5)={1,4,3}, CPC(1)={5,4}, CPC(4)={5,1,2}, CPC(2)={4}, and CPC(3)={5}.

From the trace example, we can see that when a new feature arrives, CDFSF needs to perform two conditional independence tests between the new one and each feature inside GFS in the growing phase. This is very time-consuming when the feature space is large. In order to improve the efficiency of CDFSF, we propose an S-CDFSF (symmetrical CDFSF) algorithm by plugging the symmetry check into CDFSF.

B. The S-CDFSF algorithm

According to Theorem 1, we obtain the following two properties.

Property 1 In a faithful Bayesian network, if X$\in$ PC(Y), then Y$\in$ PC(X).

Property 2 In a faithful Bayesian network, if X$\notin$ PC(Y), then Y$\notin$ PC(X).

These properties show the symmetrical relation between parents and children in a faithful Bayesian network. With these two properties, the proposed S-CDFSF algorithm uses two strategies to improve its efficiency. One strategy is to use Property 1 in the growing phase. In this phase, S-CDFSF only regards each feature inside GFS as a target T to test whether the new feature X belongs to CPC(T) while the set CPC(X) is determined according to Property 1. For example, assuming Y $\in$ GFS, when a new feature X arrives and X$\in$ CPC(Y), we directly get Y$\in$ CPC(X) and don't need to regard X as a target feature again.

The other strategy is to use Property 2 in the shrinking phase. In a similar way, if X is removed from CPC(T), then T should also be removed from CPC(X) if T is inside CPC(X)

according to Property 2. The pseudo-code of S-CDFSF is shown in Table II.

S-CDFSF performs as follows. In the growing phase, when a new feature X_i arrives, S-CDFSF selects the first feature inside GFS as a target T, then tests whether X_i belongs to its CPC(T). If so, X_i is added to CPC(T), otherwise it is discarded. If X_i is added, by Property 1, S-CDFSF also adds T into CPC(X_i).

TABLE II. THE PSEUDO-CODE OF S-CDFSF

Input: streaming features X; **Output:** CPC(X), X={X_1,X_2,…,X_n}
1. Initialization
GFS={},CPC(X)={};
2. The growing phase
(1) Generate a new feature X_i
(2) For each feature T∈ GFS, according to Corollary 2
if $\forall S \subseteq CPC(T)$ s.t Dep(X_i, T | S)
CPC(T)= CPC(T) ∪ X_i;
CPC(X_i)= CPC(X_i) ∪ T;
3.The shrinking phase
For each feature Y ∈ CPC(T), T∈ GFS
if $\exists S \subseteq CPC(T) \setminus Y$ s.t Ind(Y, T | S)
CPC(T)= CPC(T)-Y;
if T inside CPC(Y)
CPC(Y)= CPC(Y)-T
4. GFS=GFS ∪ X_i
5. Interleave step 2 to step 4 until all features arrive.

If X_i is put into CPC(T), the shrinking phase is performed. In this phase, S-CDFSF reevaluates each feature within CPC(T) and eliminates false positives from CPC(T). For example, if Y ∉ CPC(T), S-CDFSF removes it from CPC(T). According to Property 2, T should be removed from CPC(Y) without performing any tests if T is inside CPC(Y). Then S-CDFSF considers a next feature within GFS as a target until the last feature inside GFS is visited.

C. Time complexity analysis

The time complexity of the proposed two algorithms depends on the number of the conditional independence tests. Assuming N features arrive at time t, then the number of the conditional independence tests of CDFSF is approximately $O(N^2(|CPC|k^{|CPC|}+N^2k^{|CPC|}))$. Since we plug the symmetry check into S-CDFSF, the time complexity of S-CDFSF is approximately $O(N^2|CPC|k^{|CPC|})$, where k is the maximum allowable size that a conditioning set may grow and |CPC| is the largest size of the set CPC(T) over all generated features. Thus, S-CDFSF is more efficient than CDFSF.

IV. EXPERIMENTAL RESULTS

A. Experiment setup

In order to simulate the scenario of streaming features, we apply our algorithms to the traditional settings of local causal discovery, that is, those of fixed features, but the features arrive one at a time in a random way. There is no related work about causal discovery in the context of streaming features. In order to validate our algorithms, we compare them with the following existing state-of-the-art algorithms of causal discovery: HITON_PC and MMPC which are applied to standard settings where all features are available before learning (the software is available on the Web [2]). Four Bayesian networks are selected as shown in Table III. The experiments were conducted on a computer with Windows XP, 2.6GHz CPU and 2GB memory. We evaluate the algorithms using the following metrics.

(1) Precision, the number of true positives in the output divided by the number of features in the output;

(2) Recall, the number of true positives in the output divided by the number of true positives in the Bayesian network;

(3) Distance: combining precision and recall to measure the Euclidean distance from perfect precision and recall.

$$distance = \sqrt{(1-precision)^2 + (1-recall)^2}\ .$$

The conditional independence tests in our implementation are G^2 test and the parameter α is a statistical significance level for the independence tests which equals to 0.01 in our experiments.

TABLE III. SUMMARY OF SELECTED NETWORKS

Bayesian networks	*Number of variables*
Alarm	*37*
Insurance10	*270*
Child10	*200*
Gene	*801*

B. Experimental results

We run four algorithms, CDFSF, S-CDFSF, HITON_PC, and MMPC, with each feature in each Bayesian network as a target of interest, and then report the average precision, recall and distance over all features for each Bayesian network. With α up to 0.01, Fig.2 reports the experimental results by the four algorithms on the four networks with different sample sizes. Since with α up to 0.01, the performance of our two algorithms is similar with α up to 0.05, detailed results are not listed in the paper. From Fig. 2, we can draw the following conclusions.

(1) CDFSF vs. S-CDFSF. On small networks, like alarm, child10 and insurance10, S-CDFSF is highly competitive with CDFSF. On a large network, like gene, with a small sample size, the performance of CDFSF is superior to S-CDFSF. The explanation is that when the size of samples is smaller than the size of dimensions, some conditional independence tests could be unreliable. Thus, the theorem of symmetry (Property 1 and Property 2) used in the S-CDFSF algorithm could fail. But when the number of samples is much larger than the number of dimensions, S-CDFSF is highly competitive with CDFSF, even superior to CDFSF on the alarm network. Assuming that all conditional independence tests are reliable, the performance of S-CDFSF might be very competitive with CDFSF, even superior. For example, when the sample size is up to 2000, the performance of S-CDFSF is almost the same as CDFSF.

(2) Our algorithms vs. HITON_PC and MMPC. On the precision metric, our algorithms achieve a higher precision

with various sample sizes on most networks. Therefore, our algorithms usually return fewer false positives. On the recall metric, our algorithms are a little lower than the two existing algorithms, especially when the number of samples is smaller than the number of dimensions. The explanation is that in the context of streaming features, without the global information of all features, our algorithms cannot find a best candidate for each target of interest from all features at each time like HITON_PC and MMPC. But when the sample size is large, our algorithms are competitive with HITON_PC and MMPC. Combining the precision and recall metrics, our algorithms even achieve a shorter distance than the two existing algorithms on some networks.

In a word, under the assumption that all independence tests are reliable, S-CDFSF might be very competitive with CDFSF. Compared with HITON_PC and MMPC, our algorithms are a little superior to them on the precision and distance metrics. On the recall metric, our algorithms are inferior to the two algorithms when the sample size is small because of lacking global information of the feature set. But our algorithms are highly competitive with the two algorithms with large samples. Thus, we can conclude that our algorithms can well learn causal relations in the context of streaming features.

C. *Analysis of running time*

Tables IV and V show the comparisons of running time of CDFSF and S-CDFSF with α up to 0.01 and 0.05, respectively. Since the HITON_PC and MMPC algorithms were implemented in the traditional scenario while our methods were performed in the context of steaming features, the time-performance comparison between them is not conducted. In these two tables, the term "A/B" denotes that A records the running time of S-CDFSF while B is that of CDFSF. These tables depict that S-CDFSF is more efficient than CDFSF, especially with a large sample size.

TABLE IV. RUNNING TIME (SECONDS) OF S-CDFSF VS. CDFSF WITH α=0.01

#Samples	*200*	*500*	*2000*	*5000*
Alarm	*0/0*	*0/0*	*0/1*	*3/7*
Insurance10	*1/2*	*3/4*	*11/21*	*61/113*
Child10	*0/1*	*1/2*	*8/13*	*51/83*
Gene	*10/19*	*25/46*	*127/268*	*1613/3293*

TABLE V. RUNNING TIME (SECONDS) OF S-CDFSF VS. CDFSF WITH α=0.05

#Samples	*200*	*500*	*2000*	*5000*
Alarm	*0/0*	*0/0*	*1/2*	*4/9*
Insurance10	*1/2*	*3/5*	*14/27*	*53/109*
Child10	*1/1*	*1/3*	*9/17*	*67/116*
Gene	*12/22*	*32/64*	*191/410*	*2306/4668*

V. CONCLUSONS

Although there were many studies on causal discovery, there is no related work for causal discovery in the context of streaming features. In this paper, we have studied this new research problem and presented two novel algorithms. To the best of our knowledge, this is the first attempt to address causal discovery from steaming features. In order to demonstrate the effectiveness of our algorithms, we compared them with the state-of-the-art algorithms of causal discovery, HITON_PC and MMPC, which assume that all features are known in advance. The experimental results showed our methods are highly competitive with the HITON_PC and MMPC algorithms and revealed the effectiveness of our methods for the task of causal discovery with streaming features.

Meanwhile, many issues for causal discovery from streaming features remain wide open. We list here three problems which, in our opinion, deserve further investigations. Firstly, our work only identifies the essential graph of the Markov equivalence class, how to extend our work for causal structure learning in the context of streaming features needs to be further explored. Secondly, applying our algorithms to real-world data for classification is also a very interesting and essential direction, for example, feature selection on crater detection in planetary images using tens of thousands texture features. The final problem is to improve the recall metric of our algorithms on datasets of small sample sizes.

ACKNOWLEDGMENTS

This work is supported by the 973 Program of China (2009CB326203), the National Natural Science Foundation of China (60828005, 60975034 and 61070131), the US National Science Foundation (CCF-0905337), and the US NASA (NNX09AK86G).

REFERENCES

[1] C. F. Aliferis and I. Tsamardinos. Algorithms for large-scale local causal discovery and feature selection in the presence of small sample or large causal neighborhoods. *Technical Report DSL 02-08*, 2002.

[2] C. F. Aliferis, I. Tsamardinos, A. Statnikov and LE. Brown. Causal Explorer: a causal probabilistic network learning toolkit for biomedical discovery. *METMBS*, 2003.

[3] C. F. Aliferis, I. Tsamardinos and A. Statnikov. HITON: a novel Markov blanket algorithm for optimal variable selection. *AMIA 2003*, 21-25.

[4] C. F. Aliferis, A. Statnikov, I. Tsamardinos et al. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part I: Algorithms and Empirical Evaluation, *Journal of Machine Learning Research* 11: 171−234, 2010.

[5] C. F. Aliferis, A. Statnikov and I. Tsamardinos et al. Local Causal and Markov Blanket Induction for Causal Discovery and Feature Selection for Classification Part II: Analysis and Extensions, *Journal of Machine Learning Research* 11: 235−284, 2010.

[6] J. Cheng, R. Greiner and J. Kelly et al. Learning Bayesian networks from data: An information-theory based approach. *Artificial Intelligence*, 137, 43–90, 2002.

[7] D. M. Chickering, D. Heckerman, and C. Meek. Large-sample learning of Bayesian networks is NP-hard. *Journal of Machine Learning Research* 5, 1287–1330, 2004.

[8] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9(4), 309–347, 1992.

[9] N. Friedman, I. Nachman, and D. Pe'er. Learning Bayesian network structure from massive datasets: The "sparse candidate" algorithm. *UAI-99*, 1999.

[10] K. Glocer, D. Eads and J. Theiler. Online Feature Selection for Pixel Classification, *ICML*, 249-256, 2005.

[11] Kaname Kojima, Eric Perrier, Seiya Imoto, and Satoru Miyano. Optimal Search on Clustered Structural Constraint for Learning Bayesian Network Structure, *JMLR* 11, 285-310, 2010.

[12] Dan Levi and Shimon Ullman. Leaning to classify by ongoing feature seleciton, *Journal of Image and Vision Computing*, 28, 715-723, 2010.

[13] S. Mani and G. F. Cooper. Causal discovery using a Bayesian local causal discovery algorithm. *Medinfo*, 2004, 11, 731-735.

[14] D. Margaritis and S. Thrun. Bayesian network induction via local neighborhoods. *NIPS* 1999, 505-511.

[15] J. Pearl. Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, San Mateo, CA, 1988.

[16] S. Perkins and J. Theiler, Online Feature Selection Using Grafting, *ICML*, 592-599, 2003.

[17] Jean-Philippe Pellet and Andr´e Elisseeff Using Markov blankets for causal structure learning, *Journal of Machine Learning Research* 9, 1298-1342, 2008.

[18] Jose M. Peòa and R. Nilsson et al. Towards scalable and data efficient learning of Markov boundaries. *International Journal of Approximate Reasoning*, 45, 211-232, 2007.

[19] N. Slobodianik, D. Zaporozhets, N. Madras Strong Limit Theorems for the Bayesian Scoring Criterion in Bayesian Networks. *Journal of Machine Learning Research*, 10, 1511-1526, 2009.

[20] P. Spirtes, C. Glymour and R. Scheines. Causation, prediction, and search. The MIT Press, second edition, 2000.

[21] I. Tsamardinos, L. E. Brown and C. F. Aliferis. The Max-Min Hill-Climbing Bayesian Network Structure Learning Algorithm. *Machine Learning*, 65, 1:31-78, 2006.

[22] X. Wu, K. Yu, H. Wang, and W. Ding, Online Streaming Feature Selection, *ICML*, 1159-1166, 2010.

[23] J. Zhou, D. Foster, R. Stine and L. Ungar. Streamwise Feature Selection, *Journal of Machine Learning Research*, 7, 1861-1885, 2006.

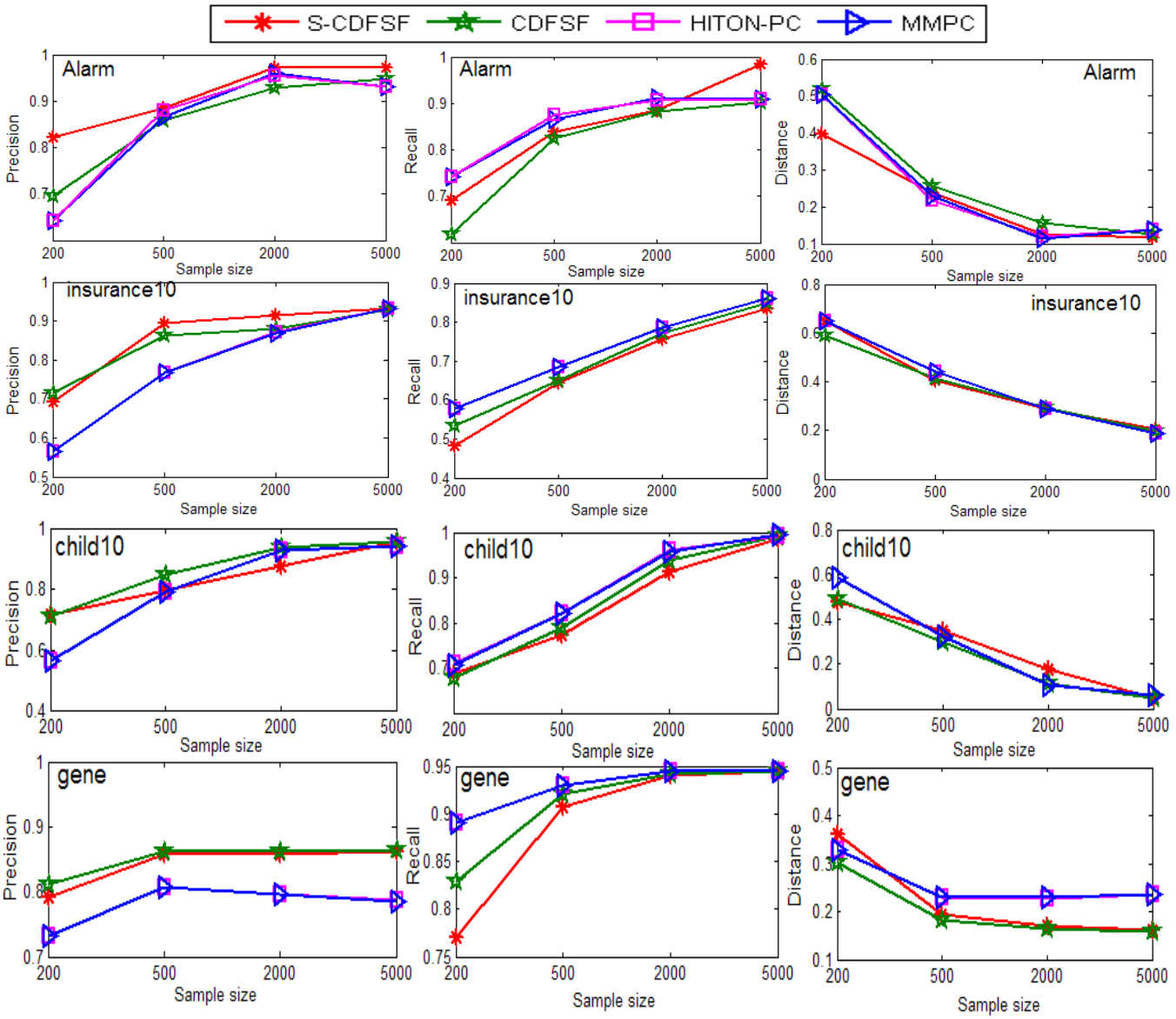

Figure 2. Experimental results for four algorithms with α=0.01