International Journal of Approximate Reasoning 159 (2023) 108945

Contents lists available at ScienceDirect

International Journal of Approximate Reasoning

journal homepage: www.elsevier.com/locate/ijar

Constraint-based and hybrid structure learning of multidimensional continuous-time Bayesian network classifiers

Carlos Villa-Blanco [a,*], Alessandro Bregoli [b], Concha Bielza [a], Pedro Larrañaga [a], Fabio Stella [b]

[a] Computational Intelligence Group, Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Madrid, Spain
[b] Models and Algorithms for Data and Text Mining, Department of Informatics, Systems and Communication, University of Milan-Bicocca, Milan, Italy

A R T I C L E   I N F O

Article history:
Received 31 January 2023
Received in revised form 16 April 2023
Accepted 16 May 2023
Available online 22 May 2023

Keywords:
Continuous-time Bayesian networks
Structure learning algorithms
Bayesian network classifiers
Multidimensional classification
Learning from data

A B S T R A C T

Learning the structure of continuous-time Bayesian networks directly from data has traditionally been performed using score-based structure learning algorithms. Only recently has a constraint-based method been proposed, proving to be more suitable under specific settings, as in modelling systems with variables having more than two states. As a result, studying diverse structure learning algorithms is essential to learn the most appropriate models according to data characteristics and task-related priorities, such as learning speed or accuracy. This article proposes alternative algorithms for learning multidimensional continuous-time Bayesian network classifiers, introducing, for the first time, constraint-based and hybrid algorithms for these models. Nevertheless, these contributions also apply to the simpler one-dimensional classification problem for which only score-based solutions exist in the literature. More specifically, the aforementioned constraint-based structure learning algorithm is first adapted to the supervised classification setting. Then, a novel algorithm of this kind, specifically tailored for the multidimensional classification problem, is presented to improve the learning times for the induction of multidimensional classifiers. Finally, a hybrid algorithm is introduced, attempting to combine the strengths of the score- and constraint-based approaches. Experiments with synthetic and real-world data are performed not only to validate the capabilities of the proposed algorithms but also to conduct a comparative study of the available competitors.

1. Introduction

The analysis and processing of time series data are present in practically any field of study, such as engineering, medicine, economics, signal processing or cybersecurity, and are essential to understanding and automating many of their processes [14,15,26,29]. These data are characterised by their large size and high dimensionality [10], which are expected to keep

* Corresponding author.
E-mail addresses: carlos.villa@upm.es (C. Villa-Blanco), a.bregoli1@campus.unimib.it (A. Bregoli), mcbielza@fi.upm.es (C. Bielza), pedro.larranaga@fi.upm.es (P. Larrañaga), fabio.stella@unimib.it (F. Stella).

https://doi.org/10.1016/j.ijar.2023.108945

growing as they become easier to collect and at higher granularity. Therefore, designing algorithms capable of modelling these data more accurately and efficiently is becoming more necessary.

In this article, we focus on learning continuous-time Bayesian network (CTBN) structures from data and, more specifically, structures of CTBNs that can be applied to the multidimensional classification of time series. CTBN classifiers have been successfully applied for one-dimensional problems such as post-stroke rehabilitation [7] or multidimensional problems like detecting energy consumption states [32]. Nevertheless, the study of structure learning algorithms for these classifiers remains largely unexplored in the literature.

The main contributions of this paper are the following:

(i) The development of the first constraint-based structure learning algorithm for CTBN classifiers (CTBNCs), which is conceived to learn multidimensional continuous-time Bayesian network classifiers (Multi-CTBNCs).
(ii) The introduction of the first hybrid structure learning algorithm for Multi-CTBNCs.
(iii) A comprehensive comparative study to evaluate the strengths and weaknesses of the state-of-the-art algorithms and those proposed in this work.
(iv) The presentation of a multidimensional classification problem that uses publicly available data to demonstrate the usefulness of the proposed constraint-based algorithm in a real-world setting.
(v) The development and integration of the presented algorithms into a software tool introduced in [32].

The remainder of this article is as follows. Section 2 reviews fundamental concepts. Sections 3 and 4 introduce novel constraint-based and hybrid algorithms for Multi-CTBNCs, respectively. Section 5 presents experiments and discusses the results of multiple structure learning algorithms. Section 6 concludes the article and discusses future research lines.

2. Fundamentals

A Bayesian network (BN) is a probabilistic graphical model (PGM) designed for reasoning about static processes [22]. Thus, such a model is unsuitable when a system exhibits temporal behaviour. For this reason, the CTBN has been proposed to represent the temporal dynamics of continuous-time and discrete-state stochastic processes [20], which are described by finite state, continuous-time homogeneous Markov processes through intensity matrices.

Definition 1 *(Homogeneous Markov process).* Given a discrete random variable X, whose sample space is $\Omega_X = \{x_1, \dots, x_k\}$, its transient behaviour can be described as a homogeneous Markov process $X(t)$ with its intensity matrix:

$$\mathbf{Q}_X = \begin{bmatrix} -q_{x_1} & q_{x_1,x_2} & \cdots & q_{x_1,x_k} \\ q_{x_2,x_1} & -q_{x_2} & \cdots & q_{x_2,x_k} \\ \vdots & \dots & \ddots & \vdots \\ q_{x_k,x_1} & q_{x_k,x_2} & \cdots & -q_{x_k} \end{bmatrix},$$

where q_{x_a,x_b} is the intensity of leaving state x_a and arriving at x_b and $q_{x_a} = \sum_{b \neq a} q_{x_a,x_b}$ is the intensity of variable X leaving state x_a. The waiting time of variable X in a given state x_a is exponentially distributed with parameter q_{x_a}. Thus, variable X is expected to transition from state x_a at time $1/q_{x_a}$ and to a state x_b with probability $q_{x_a,x_b}/q_{x_a}$ [20].

When modelling systems with multiple variables $\mathcal{X} = \{X_1, \dots, X_m\}$, a simple solution would be to define an intensity matrix over the joint sample space of $\mathcal{X}$. However, this approach would only be feasible with a very limited number of variables, as the size of the intensity matrix would grow exponentially with their number and cardinality. Therefore, in order to model larger systems, a factored representation of Markov processes, known as conditional Markov process, should be employed.

Definition 2 *(Conditional Markov process).* A conditional Markov process is a type of inhomogeneous Markov process whose intensity matrix changes over time as a function of some conditioning variables' state. Given a discrete random variable X_i and a set of parents $\mathbf{Pa}(X_i)$ of X_i in a directed graph $\mathcal{G}$, a conditional intensity matrix (CIM) $\mathcal{Q}_{X_i}^{\mathbf{Pa}(X_i)}$ describes the temporal dynamics of the variable. A CIM is a set of homogeneous intensity matrices $\mathbf{Q}_{X_i}^{\mathbf{pa}(X_i)}$, each encoding the dynamics of X_i given the state $\mathbf{pa}(X_i)$ of its parents $\mathbf{Pa}(X_i)$:

$$\mathbf{Q}_{X_i}^{\mathbf{pa}(X_i)} = \begin{bmatrix} -q_{x_1}^{\mathbf{pa}(X_i)} & q_{x_1,x_2}^{\mathbf{pa}(X_i)} & \cdots & q_{x_1,x_k}^{\mathbf{pa}(X_i)} \\ q_{x_2,x_1}^{\mathbf{pa}(X_i)} & -q_{x_2}^{\mathbf{pa}(X_i)} & \cdots & q_{x_2,x_k}^{\mathbf{pa}(X_i)} \\ \vdots & \dots & \ddots & \vdots \\ q_{x_k,x_1}^{\mathbf{pa}(X_i)} & q_{x_k,x_2}^{\mathbf{pa}(X_i)} & \cdots & -q_{x_k}^{\mathbf{pa}(X_i)} \end{bmatrix}.$$

CIMs can be summarised with the sets of parameters $q_{x_a}^{\mathbf{pa}(X_i)}$ and $\theta_{x_a,x_b}^{\mathbf{pa}(X_i)} = q_{x_a,x_b}^{\mathbf{pa}(X_i)} / q_{x_a}^{\mathbf{pa}(X_i)}$, the latter defining the probability of transitioning from state x_a to another x_b when a transition is known to occur.

CTBNs describe the temporal dynamics of some discrete random variables and their dependencies on each other using conditional intensity matrices while providing a graphical representation of the modelled system. CTBNs have been successfully applied in a variety of real-world problems, including intrusion detection [33], gene network reconstruction [1], speech-related facial action unit recognition [19] or disease monitoring [18], among others.

Definition 3 *(Continuous-time Bayesian network).* A CTBN $\mathbb{N} = (\mathcal{G}, \mathcal{Q}, P^0_{\mathcal{X}})$ over a set of discrete random variables $\mathcal{X} = \{X_1, \ldots, X_m\}$ consists of:

- A continuous transition model specified by a directed (possibly cyclic) graph $\mathcal{G}$ over $\mathcal{X}$ and a CIM $\mathcal{Q}^{\mathbf{Pa}(X_i)}_{X_i}$ for each variable X_i.
- An initial distribution $P^0_{\mathcal{X}}$, specified as a BN over $\mathcal{X}$, representing the initial state of a temporal process.

CTBNs can be applied not only in knowledge discovery but also to perform classification tasks, which is the focus of this work. CTBNCs, introduced by Stella and Amer [25], extend CTBNs to classify discrete-state temporal sequences $\mathcal{S}_l = \{\mathbf{x}_l^{t_1}, \ldots, \mathbf{x}_l^{t_{T_l}}, c_l\}$ $(l = 1, \ldots, N)$,[1] where N is the number of sequences, which describe the transitions of feature variables $\mathcal{X}$ (with values $\mathbf{x}_l^{t_j}$) and the state of a single class variable C (with values c_l) that does not depend on time.

Definition 4 *(Continuous-time Bayesian network classifier).* Given a set of discrete random variables $\mathcal{V} = \{X_1, \ldots, X_m, C\}$, a CTBNC is a pair $\mathbb{C} = (\mathbb{N}, P(C))$, where $\mathbb{N}$ is a CTBN over time-dependent feature variables $\mathcal{X} = \{X_1, \ldots, X_m\}$ and C is a time-independent class variable fully specified by the marginal probability $P(C)$ on states $\Omega_C = \{c_1, \ldots, c_r\}$. The graph of a CTBNC has the same properties as that of a CTBN but includes a class variable node with no parents, i.e., $\mathbf{Pa}(C) = \emptyset$.

Certain classification problems require predicting simultaneously the state of multiple class variables $\mathcal{C} = \{C_1, \ldots, C_d\}$, i.e., $\mathcal{S}_l = \{\mathbf{x}_l^{t_1}, \ldots, \mathbf{x}_l^{t_{T_l}}, \mathbf{c}_l\}$, where d is the number of class variables and $\mathbf{c}_l = (c_{l1}, \ldots, c_{ld})$. One classifier can be learned for each class variable to solve this problem. Nevertheless, this approach would fail to identify inter-class dependencies, which could provide relevant information for the classification. In this more complex setting, using a model such as the Multi-CTBNC, introduced in [32], is a more appropriate alternative since it captures the probabilistic relationships of conditional (in)dependence among class variables.

Definition 5 *(Multidimensional continuous-time Bayesian network classifier).* A Multi-CTBNC $\mathbb{M} = (\mathcal{G}, \mathbf{B}, \mathcal{Q}, P^0_{\mathcal{V}})$ over a set of discrete random variables $\mathcal{V} = \{X_1, \ldots, X_m, C_1, \ldots, C_d\}$ is formed by:
- A directed (possibly cyclic) graph $\mathcal{G} = (\mathcal{V}, \mathcal{A})$, where vertices $\mathcal{V}$ are divided into those for feature and class variables, while arcs are between class variables (class subgraph), feature variables (feature subgraph) and from class to feature variables (bridge subgraph).
- Class variable parameters $\mathbf{B}$, which form conditional probability tables (CPTs), associated with the class subgraph.
- A set of CIMs $\mathcal{Q}^{\mathbf{Pa}(X_i)}_{X_i}$, one for each feature variable X_i.
- An initial distribution $P^0_{\mathcal{V}}$, specified as a multidimensional BN classifier over $\mathcal{V}$.

2.1. Structure learning algorithms for CTBNs

Structure learning of CTBNs has been traditionally addressed as an optimisation problem [7,21,31], where a structure is selected from a candidate space by maximising a score (score-based algorithms). The motivation may be that standard score-based algorithms can be straightforwardly applied to learn CTBNs. As CTBNs have no acyclicity constraints, the parent set of each node can even be conveniently defined in parallel without concern for reporting erroneous structures. Only recently has a constraint-based algorithm been proposed, which reconstructs their structures by performing conditional independence tests. The continuous-time PC (CTPC) algorithm, introduced by Bregoli et al. [6], is the first proposal of this kind, which adapts the classical PC algorithm, briefly described in Algorithm 1 (for a more detailed explanation, see [24] and [8]), to CTBNs. As CIMs describe temporal dynamics, classical statistical tests cannot be applied. Thus, CTPC introduces a novel definition of conditional independence in CTBNs.

Definition 6 *(Conditional independence in CTBNs).* Given a CTBN over a set of discrete random variables $\mathcal{X} = \{X_1, \ldots, X_m\}$, a variable X_i is conditionally independent of X_j given a separating set $\mathbf{S}_{X_iX_j} \subseteq \mathcal{X} \setminus \{X_i, X_j\}$ iff:

$$\mathbf{Q}^{y,\mathbf{s}}_{X_i} = \mathbf{Q}^{\mathbf{s}}_{X_i} \qquad \forall y \in \Omega_{X_j}, \forall \mathbf{s} \in \Omega_{\mathbf{S}_{X_iX_j}}.$$

[1] Sequences may have different timestamps; superscript l will be omitted from t for simplicity.

Algorithm 1 PC($\mathcal{V}$).

```
1: Build the complete undirected graph $\mathcal{G}$ on node set $\mathcal{V}$
2: Find the skeleton and separating sets of $\mathcal{G}$. For each pair of adjacent nodes $V_i, V_j \in \mathcal{V}$, remove edge $V_i - V_j$ from $\mathcal{G}$ iff there is a separating set $\mathbf{S}_{V_iV_j} \subseteq \mathcal{V} \setminus \{V_i, V_j\}$ such that $V_i \perp\!\!\!\perp V_j | \mathbf{S}_{V_iV_j}$
3: Orient the colliders using the separating sets. For any path $V_i - V_j - V_k$ in $\mathcal{G}$, such that $V_i \not- V_k$, orient the edges as $V_i \to V_j \leftarrow V_k$ iff $V_i \perp\!\!\!\perp V_k | \mathbf{S}_{V_iV_k}$ and $V_j \notin \mathbf{S}_{V_iV_k}$
4: Orient all possible undirected edges:
  4.1: Given that $V_i \to V_j - V_k$ and $V_i \not- V_k$, then orient the undirected edge into $V_j \to V_k$ to avoid introducing a new v-structure
  4.2: Given that $V_i - V_k$ and $V_i \to V_j \to V_k$, then orient $V_i - V_k$ into $V_i \to V_k$ to avoid introducing a cycle
  4.3: Given that $V_i - V_k$, $V_i - V_j \to V_k$, $V_i - V_w \to V_k$ and $V_i \not- V_w$, then orient $V_i - V_k$ into $V_i \to V_k$ to avoid introducing a new v-structure or a cycle
5: return partially directed acyclic graph $\mathcal{G}$
```

Definition 6 is not symmetric, which means that while a variable X_i could be conditionally independent of X_j given $\mathbf{S}_{X_iX_j}$, the opposite does not necessarily have to be true. This fact has an important implication in learning the structure of a CTBN since it is necessary to evaluate twice as many arcs with respect to a traditional BN.

The CTPC algorithm employs two different statistical tests to establish conditional independence between some feature variables X_i and X_j. First, the time-to-transition null hypothesis (Definition 7) is evaluated to determine if significant differences exist for the waiting times of variable X_i when X_j is added (or not) to its parents. Similarly, the state-to-state-transition null hypothesis (Definition 8) is subsequently evaluated for significant differences in the probabilities of variable X_i transitioning from one particular state to another. Conditional independence is not established if any null hypothesis is rejected.

Definition 7 *(Time-to-transition null hypothesis).* Given two variables X_i and X_j, and a separating set $\mathbf{S}_{X_iX_j} \subseteq \mathcal{X} \setminus \{X_i, X_j\}$, the time-to-transition null hypothesis of X_j over X_i is defined as:

$$q_x^{y,\mathbf{s}} = q_x^{\mathbf{s}} \qquad \forall x \in \Omega_{X_i}, \forall y \in \Omega_{X_j}, \forall \mathbf{s} \in \Omega_{\mathbf{S}_{X_iX_j}}.$$

Definition 8 *(State-to-state-transition null hypothesis).* Given two variables X_i and X_j, and a separating set $\mathbf{S}_{X_iX_j} \subseteq \mathcal{X} \setminus \{X_i, X_j\}$, the state-to-state-transition null hypothesis of X_j over X_i is defined as:

$$\theta_{x_a,x_b}^{y,\mathbf{s}} = \theta_{x_a,x_b}^{\mathbf{s}} \qquad \forall x_a \in \Omega_{X_i}, \forall x_b \in \Omega_{X_i} \setminus \{x_a\}, \forall y \in \Omega_{X_j}, \forall \mathbf{s} \in \Omega_{\mathbf{S}_{X_iX_j}}.$$

Bregoli et al. [6] proposed to test the time-to-transition null hypothesis using the F-test, while they explored the use of the two-sample chi-square and Kolmogorov-Smirnov tests for the state-to-state-transition null hypothesis. Nevertheless, they found the two-sample chi-square test to be marginally better for testing the latter null hypothesis, so only this test will be employed in the present work.

The CTPC algorithm can be easily adapted to learn the bridge and feature subgraphs of Multi-CTBNCs since only the parent sets of nodes with CIMs are learned. The algorithm only has to meet the topology constraints of the model. Algorithm 2 shows the pseudocode of the CTPC algorithm used in this work, where $\mathcal{V} = \mathcal{X} \cup \mathcal{C}$ (see Definition 5). As for learning the class subgraph, traditional constraint-based solutions for discrete BNs can be used. We call this adaption "naive" since it tests all possible dependencies between feature variables, even those irrelevant to the classification task. This considerably increases the learning time, a problem aggravated in CTBNs since dependencies between feature variables are non-symmetric.

Algorithm 2 CTPC($\mathcal{X}$, $\mathcal{V}$)[a].

```
1: for each feature variable $X_i \in \mathcal{X}$ do
2:     Set $\mathcal{U} = \{V_j \in \mathcal{V} | V_j \to X_i\}$
3:     for increasing values $s = 0, 1, \ldots, |\mathcal{U}| - 1$ do
4:         for each variable $V_j \in \mathcal{U}$ and subset $\mathbf{S}_{X_iV_j} \subseteq \mathcal{U} \setminus \{V_j\}$, where $|\mathbf{S}_{X_iV_j}| = s$ do
5:             if $X_i \perp\!\!\!\perp V_j | \mathbf{S}_{X_iV_j}$ then
6:                 Remove arc $V_j \to X_i$ from $\mathcal{G}$ and delete $V_j$ from $\mathcal{U}$
7:             end if
8:         end for
9:     end for
10: end for
11: return directed graph $\mathcal{G}$
```

[a] Adaptation of the algorithm by Bregoli et al. [6]. The first step of the original algorithm, which forms a complete directed graph, was omitted for convenience.

Algorithm 3 MB-CTPC($\mathcal{X}$, $\mathcal{C}$).

```
1: $\mathcal{G} \leftarrow$ PC($\mathcal{C}$)
2: Build the complete bridge and feature subgraphs of $\mathcal{G}$ on node set $\mathcal{X} \cup \mathcal{C}$
3: $\mathcal{G} \leftarrow$ CTPC($\mathcal{X}$, $\mathcal{C}$)
4: for each feature variable $X_i \in \mathcal{X}$ do
5:     for each feature variable $X_j \in \mathcal{X}$, where $X_j \neq X_i$ do
6:         if $\mathbf{Pa}_{\mathcal{C}}(X_i) = \emptyset$ then
7:             Remove arc $X_j \rightarrow X_i$ from $\mathcal{G}$
8:         else if $\mathbf{Pa}_{\mathcal{C}}(X_i) \cap \mathbf{Pa}_{\mathcal{C}}(X_j) = \emptyset$ AND $\mathbf{Pa}_{\mathcal{C}}(X_j) \neq \emptyset$ then
9:             Remove arcs $X_i \rightarrow X_j$ and $X_j \rightarrow X_i$ from $\mathcal{G}$
10:        else if $\mathbf{Pa}_{\mathcal{C}}(X_i) \setminus \mathbf{Pa}_{\mathcal{C}}(X_j) \neq \emptyset$ AND $\mathbf{Pa}_{\mathcal{C}}(X_j) \neq \emptyset$ then
11:            Remove arc $X_i \rightarrow X_j$ from $\mathcal{G}$
12:        end if
13:    end for
14: end for
15: $\mathcal{G} \leftarrow$ CTPC($\mathcal{X}$, $\mathcal{C} \cup \mathcal{X}$)
16: return directed graph $\mathcal{G}$
```

3. Markov blanket-based continuous-time PC algorithm

This section introduces a novel constraint-based structure learning algorithm called Markov blanket-based continuous-time PC (MB-CTPC), specially designed to learn Multi-CTBNCs. This algorithm aims to evaluate only those dependencies relevant to the Markov blanket[2] of the class variables, which provides sufficient information to infer these variables. To this end, MB-CTPC defines a set of rules to ignore irrelevant dependencies based on ancestor class variables of feature variables.

Algorithm 3 describes the pseudocode of MB-CTPC. Step 1 finds the probabilistic relationships between class variables using a traditional constraint-based algorithm, such as PC. Step 2 forms the complete bridge and feature subgraphs of $\mathcal{G}$. Then, Step 3 defines the descendants of the class variables via conditional independence tests between feature and class variables without considering other feature variables in the separating set. Thus, a dependency of a feature on a class variable might exist because it is its child or there is a flow of information through intermediate feature nodes. This step defines a preliminary bridge subgraph that provides valuable information to reduce the statistical tests for the feature subgraph. If a pair of feature variables do not share the same parent class variables, information is not flowing in at least one direction, and at least one dependency can be removed. Steps 4 to 14 use three rules to reduce the number of conditional independence tests:

Rule 1 (Steps 6 and 7). Given adjacent feature variables X_i and X_j, arc $X_i \rightarrow X_j$ is removed iff $\mathbf{Pa}_{\mathcal{C}}(X_j) = \emptyset$, where $\mathbf{Pa}_{\mathcal{C}}(X_j)$ denotes the parent class variables of X_j.

Rule 2 (Steps 8 and 9). Given adjacent feature variables X_i and X_j, arcs $X_i \rightarrow X_j$ and $X_j \rightarrow X_i$ are removed iff $\mathbf{Pa}_{\mathcal{C}}(X_i) \cap \mathbf{Pa}_{\mathcal{C}}(X_j) = \emptyset$, $\mathbf{Pa}_{\mathcal{C}}(X_i) \neq \emptyset$ and $\mathbf{Pa}_{\mathcal{C}}(X_j) \neq \emptyset$.

Rule 3 (Steps 10 and 11). Given adjacent feature variables X_i and X_j, arc $X_i \rightarrow X_j$ is removed iff $\mathbf{Pa}_{\mathcal{C}}(X_i) \setminus \mathbf{Pa}_{\mathcal{C}}(X_j) \neq \emptyset$, $\mathbf{Pa}_{\mathcal{C}}(X_i) \neq \emptyset$ and $\mathbf{Pa}_{\mathcal{C}}(X_j) \neq \emptyset$.

Finally, Step 15 further identifies conditional independence relationships using the CTPC algorithm. However, its execution time is significantly reduced, as it is limited to evaluating only those arcs the previous rules could not discard.

Example 1. Given some data sampled from the Multi-CTBNC of Fig. 1a, then Figs. 1b to 1h show the steps of MB-CTPC to learn the Markov blankets of the class variables. First, Fig. 1b represents the learning of the class subgraph (Step 1). Then, the complete bridge and feature subgraphs are built in Fig. 1c (Step 2). Afterwards, conditional independence tests, which consider only class variables in the separating sets, find the descendant feature variables of the class variables (Step 3). For example, Fig. 1d shows an arc from C_4 to X_6, as a direct path exists through X_7 in the original structure. Subsequently, Rule 1 removes incoming arcs of X_2 and X_5 in Fig. 1e since they have no dependencies on class variables. Then, Rule 2 discards dependencies between feature variables not sharing parent class variables. That is the case for pairs like X_3 and X_6 or X_1 and X_7 in Fig. 1f. In Fig. 1g, Rule 3 removes a dependency from one feature variable to another if the former has parent class variables that the latter does not, such as arcs (X_1, X_3) and (X_6, X_7). These rules discard 27 out of 42 arcs (64%) of the feature subgraph without performing any conditional independence test. Finally, tests are performed on the remaining arcs in Fig. 1h. The resulting structure of Fig. 1i shows that arcs from the original structure providing no information about the Markov blankets of class variables (those between X_2 and X_5) are discarded. ■

4. Hybrid structure learning algorithm

Score- and constraint-based algorithms have their own strengths and weaknesses, making them more or less useful in different classification contexts. Therefore, as already done for other PGMs [2,16,27,28], we study hybrid algorithms for Multi-CTBNCs. These methods aim to combine the advantages of both approaches, such as faster learning speed of

[2] Given a Multi-CTBNC, the Markov blanket of a node consists of its parents, children and spouses.

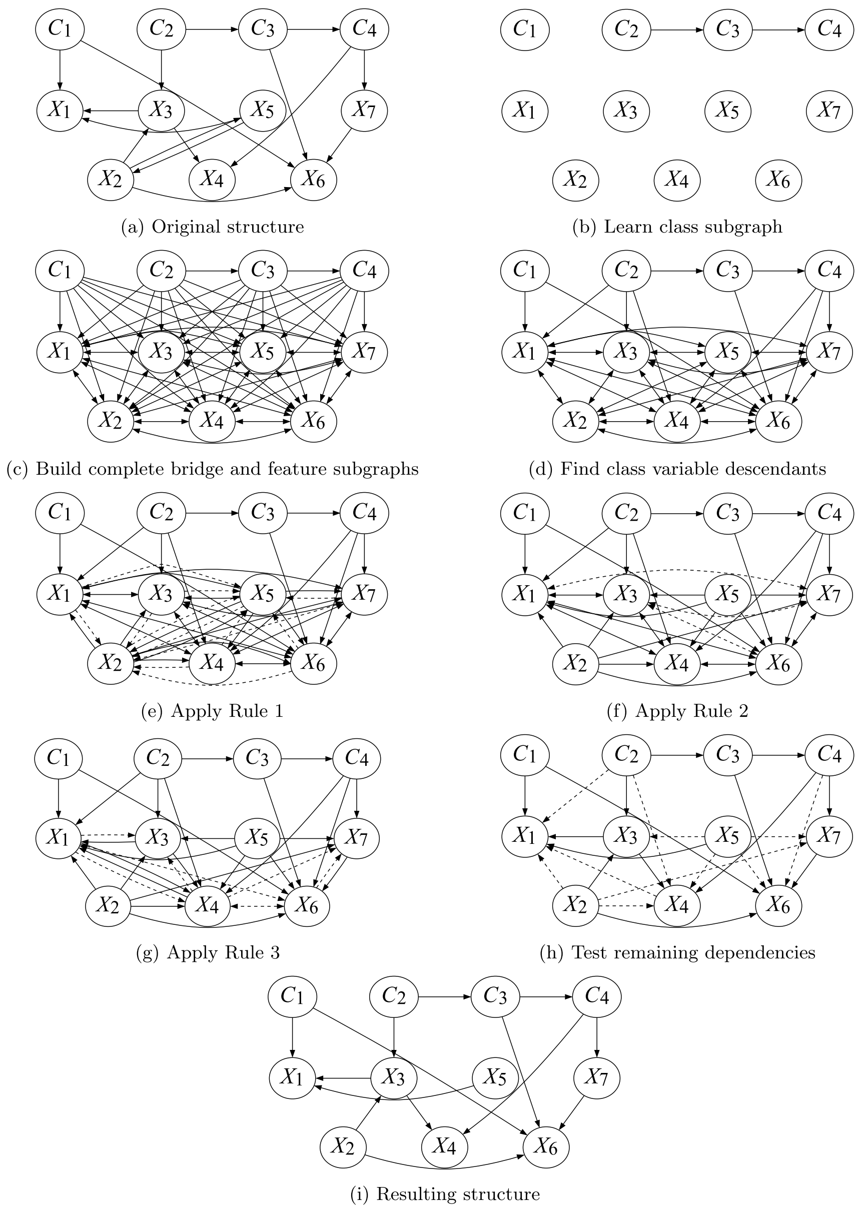


Fig. 1. Steps of the MB-CTPC algorithm. Dash lines represent removed arcs.

constraint-based algorithms and higher accuracy of score-based solutions (see Section 5.2.2). This section presents the first hybrid algorithm to learn both one-dimensional and multidimensional CTBNCs. The algorithm is divided into a restriction phase where conditional independence tests find an initial structure and a maximisation phase that refines it. Two variants are used depending on the subgraph:

- **Class subgraph**: the PC algorithm is used to reconstruct the skeleton of the class subgraph. Then, a hill climbing procedure searches for a solution, starting from the empty subgraph but only allowing arcs included in the skeleton.
- **Bridge and feature subgraphs**: the CTPC algorithm defines an initial structure during the restriction phase, which serves as the initial solution for a hill climbing algorithm in the maximisation phase. Two aspects balance the influence of these algorithms. First, a maximum separating set size is established for conditional independence tests. Second, hill climbing only removes or adds arcs that the restriction phase has not discarded. The maximum size of the separating set dictates each algorithm's influence and range of action.

Although the scope of this article is learning classifiers, this hybrid algorithm can also be applied to learn CTBNs, making it their first hybrid proposal to the best of our knowledge.

5. Experiments

This section empirically compares the performance of Multi-CTBNCs learned with five different structure learning algorithms in a variety of contexts, including synthetic datasets and a real-world problem. Score-based algorithms are represented by hill climbing and tabu search (tabu list of size 5), whose scores (BIC or BDe [32]) are indicated in square brackets, e.g., hill climbing [BDe]. Regarding constraint-based algorithms, (the naive) CTPC and the presented MB-CTPC are evaluated. Significance levels of 0.05 (class subgraph) and 1e−5[3] (bridge and feature subgraphs) are used to test conditional independence. Finally, the proposed hybrid algorithm is studied using the hill climbing [BIC] algorithm in the maximisation phase and separating sets of zero (hybrid $[|\mathbf{S}_{V_iV_j}|=0]$) and one (hybrid $[|\mathbf{S}_{V_iV_j}|=1]$) maximum size.

In order to guarantee an honest and fair comparison, the learned models are evaluated using several performance measures and a 5-fold cross-validation scheme. The measures under consideration are global and mean accuracy, global Brier score, macro and micro F_1 score and learning and classification time. For the comparisons, the Wilcoxon signed-rank test is used with a significance level of 0.05 to verify that the results are statistically significant. Regarding parameter learning, Bayesian estimation is used with the following hyperparameters for their prior distributions (for more details, see [32]): $\lambda_{c_j}=1$ and $\alpha_{x_a,x_b}=1$ for the Dirichlet prior distribution, and $\tau_{x_j}=0.001$ (synthetic experiments) or $\tau_{x_j}=1$ (real-world experiment) for the gamma prior distribution.

The experiments were run on a 4.20 GHz Intel Core i7-7700 K with 32 GB of RAM using Windows 10. The structure learning algorithms were developed in Java, and the software and datasets are freely available at https://github.com/carlvilla/Multi-CTBNCs.

5.1. Performance measures

The following measures will be used to compare the performance of Multi-CTBNCs learned with different structure learning algorithms:

- Global accuracy [4]: ratio of sequences correctly classified for all class variables, i.e., a partially correct or completely incorrect classification is considered an error:

$$Acc=\frac{1}{N}\sum_{l=1}^{N}\delta(\mathbf{c}'_l,\mathbf{c}_l),$$

  where $\mathbf{c}'_l$ and $\mathbf{c}_l$ are the predicted and actual classes of sequence l, respectively, and $\delta(\cdot,\cdot)$ is the Kronecker's delta function, so $\delta(\mathbf{c}'_l,\mathbf{c}_l)=1$ if $\mathbf{c}'_l=\mathbf{c}_l$ and 0 otherwise.
- Mean accuracy [4]: mean of the accuracies obtained for each class variable separately:

$$\overline{Acc}=\frac{1}{d}\sum_{i=1}^{d}Acc_i=\frac{1}{d}\sum_{i=1}^{d}\frac{1}{N}\sum_{l=1}^{N}\delta(c'_{li}c_{li}),$$

  where Acc_i is the accuracy for class variable C_i and N is the number of sequences.
- Global Brier score [9]: it measures the accuracy of probabilistic classifiers by considering the probability that assigns to multidimensional predictions:

$$Bs=\frac{1}{N}\sum_{l=1}^{N}\sum_{g=1}^{|\mathcal{I}|}\left(p(\mathbf{C}=\mathbf{c}_g|\mathbf{x}_l^{t_1},\ldots,\mathbf{x}_l^{t_{T_l}})-\delta(\mathbf{c}_g,\mathbf{c}_l)\right)^2,$$

  where $\mathcal{I}=\Omega_{C_1}\times\cdots\times\Omega_{C_d}$ is the space of joint configurations of the class variables. The closer this measure is to zero, the more accurate the classifier will be.
- F_1 score: harmonic mean of the precision (P) and recall (R) on a class variable C with class labels c:

$$\text{F}_1=2\frac{PR}{P+R}=2\frac{tp_c}{2tp_c+fp_c+fn_c},$$

  where tp_c, fp_c and fn_c are the counts for true positives, false positives and false negatives, respectively, for class c.

Traditional equations for precision, recall and, therefore, F_1 score can only be used for a unique binary class variable. Let B be a function that computes any of these performance measures from a confusion matrix, then the measures are obtained as follows:

[3] The large number of hypothesis tests used to compare parameters from exponential distributions requires a very small significance level to avoid inaccurately dense structures.

Table 1
Parameters used to generate the datasets for the experiments.

Parameter	Studied values
Number of feature variables	5, 10, 20
Cardinality of feature variables	2, 3, 4, 8
Number of class variables	4
Cardinality of class variables	2, 3
Density of class subgraph	30%
Density of bridge subgraph	5%, 10%, 20%
Density of feature subgraph	5%, 10%, 20%

- Macro-averaging [12]: averages the scores of each class variable C_i:

$$B_{macro} = \frac{1}{d}\sum_{i=1}^{d} B_{C_i}, \text{where } B_{C_i} = \begin{cases} \frac{1}{|\Omega_{C_i}|}\sum_{c_i} B(tp_{c_i}, fp_{c_i}, tn_{c_i}, fn_{c_i}) & \text{if } |\Omega_{C_i}| > 2 \\ B(tp_{C_i}, fp_{C_i}, tn_{C_i}, fn_{C_i}) & \text{otherwise,} \end{cases}$$

  and tn_{c_i} is the counts for true negatives for class c_i. Note that if the class variable C_i is binary, only the confusion matrix for one of its classes $(tp_{C_i}, fp_{C_i}, tn_{C_i}, fn_{C_i})$ is considered.
- Micro-averaging [12]: aggregates the confusion matrices of each class variable:

$$B_{micro} = B(\sum_{i=1}^{d} TP_{C_i}, \sum_{i=1}^{d} FP_{C_i}, \sum_{i=1}^{d} TN_{C_i}, \sum_{i=1}^{d} FN_{C_i}),$$

  where

$$\{TP_{C_i}, FP_{C_i}, TN_{C_i}, FN_{C_i}\} = \begin{cases} \frac{1}{|\Omega_{C_i}|}\sum_{c_i}\{tp_{c_i}, fp_{c_i}, tn_{c_i}, fn_{c_i}\} & \text{if } |\Omega_{C_i}| > 2 \\ \{tp_{C_i}, fp_{C_i}, tn_{C_i}, fn_{C_i}\} & \text{otherwise.} \end{cases}$$

These two averaging methods will be used in our experiments to calculate the F_1 score.[4] However, since the micro-averaging approach is equivalent to the mean accuracy when the cardinality of all class variables is the same and greater than two [32], it will only be considered in experiments where class variables have different cardinalities.

5.2. Experimental results on synthetic data

Due to the limited progress in the literature on the classification problem under study, publicly available real-world datasets are scarce. Thus, this section uses synthetic datasets drawn from various contexts to evaluate structure learning algorithms' advantages and disadvantages effectively.

Synthetic datasets are sampled via probabilistic logic sampling [13] from Multi-CTBNCs whose structures and parameters are randomly generated. Five datasets have been sampled from each combination of parameters' values (216 combinations) shown in Table 1 (1080 datasets), each with 5000 sequences that last 20 time units. The generated structures have at least one arc in the bridge subgraph, and feature variables are restricted to a maximum of three children to avoid memory problems.

Table 2 presents the results for all datasets of specific comparisons of structure learning algorithms that we found most relevant, whereas Table 3 includes the average results for each algorithm. In the following sections, we discuss some conclusions drawn from these tables and perform a more exhaustive analysis.

5.2.1. Hill climbing and tabu search

Table 2 shows that hill climbing optimising the BIC score obtains better results in all performance measures, except learning and classification time, for more datasets than the BDe score, improvements that were found statistically significant. As a result, subsequent comparisons will mainly consider the BIC score. Meanwhile, learning and classification time differences between hill climbing [BIC] and tabu search [BIC] were statistically significant, with the latter being faster in more than 60% of the datasets, while no statistically significant differences were detected for other performance measures. The reason is that, thanks to the restrictions on the structure search imposed by the tabu list, the tabu search algorithm finds the structure by analysing fewer arcs. However, note that these time differences may not be very substantial since tabu search [BIC] achieves a sub-second improvement in 60% and 96% of the cases in terms of learning and classification time, respectively.

[4] Henceforth referred to as macro or micro F_1 score for convenience.

Table 2
Percentage of datasets in which structure learning algorithms achieve better, worse or identical results.

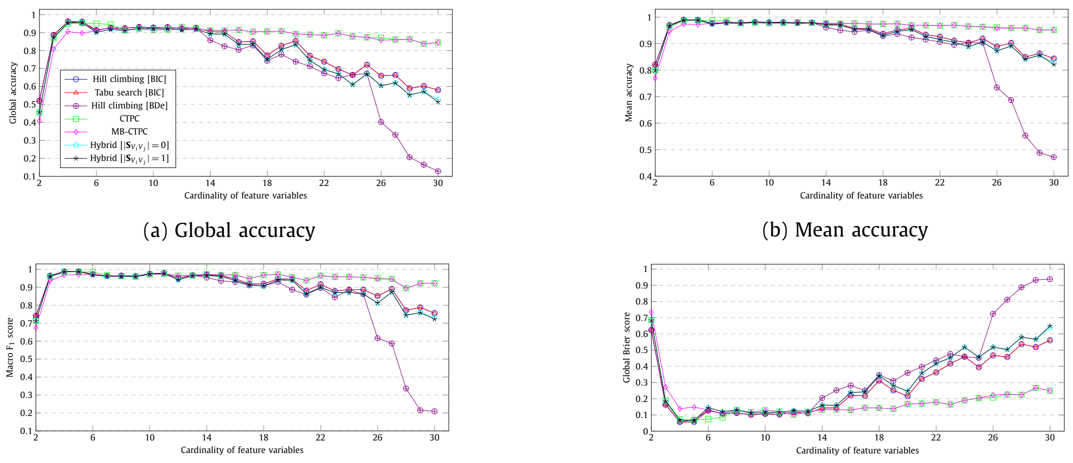
Performance measure	Results of ... are	Better	Worse	Same	Same or Better	Than
Global accuracy	Hill climbing [BIC]	10.56%	5.93%	**83.52%**	94.07%	Hill climbing [BDe]
	Hill climbing [BIC]	1.20%	0.83%	**97.96%**	99.17%	Tabu search [BIC]
	CTPC	17.69%	**42.31%**	40.00%	57.69%	Hill climbing [BIC]
	MB-CTPC	1.76%	39.26%	**58.98%**	60.74%	CTPC
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	1.76%	0.00%	**98.24%**	100.00%	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	5.56%	34.54%	**59.91%**	65.46%	Hill climbing [BIC]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	22.96%	20.56%	**56.48%**	79.44%	CTPC
Mean accuracy	Hill climbing [BIC]	11.39%	6.48%	**82.13%**	93.52%	Hill climbing [BDe]
	Hill climbing [BIC]	1.20%	1.20%	**97.59%**	98.80%	Tabu search [BIC]
	CTPC	21.39%	**39.44%**	39.17%	60.56%	Hill climbing [BIC]
	MB-CTPC	1.67%	40.28%	**58.06%**	59.72%	CTPC
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	1.85%	0.00%	**98.15%**	100.00%	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	9.35%	31.85%	**58.80%**	68.15%	Hill climbing [BIC]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	22.59%	21.76%	**55.65%**	78.24%	CTPC
Macro F_1 score	Hill climbing [BIC]	12.13%	7.59%	**80.28%**	92.41%	Hill climbing [BDe]
	Hill climbing [BIC]	1.30%	1.30%	**97.41%**	98.70%	Tabu search [BIC]
	CTPC	22.41%	**41.11%**	36.48%	58.89%	Hill climbing [BIC]
	MB-CTPC	2.31%	40.28%	**57.41%**	59.72%	CTPC
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	1.85%	0.00%	**98.15%**	100.00%	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	9.91%	33.24%	**56.85%**	66.76%	Hill climbing [BIC]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	23.33%	22.78%	**53.89%**	77.22%	CTPC
Global Brier score	Hill climbing [BIC]	24.81%	18.89%	**56.30%**	81.11%	Hill climbing [BDe]
	Hill climbing [BIC]	5.19%	5.28%	**89.54%**	94.72%	Tabu search [BIC]
	CTPC	19.91%	**59.81%**	20.28%	40.19%	Hill climbing [BIC]
	MB-CTPC	11.85%	**53.52%**	34.63%	46.48%	CTPC
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	3.52%	2.59%	**93.89%**	97.41%	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	7.41%	**49.26%**	43.33%	50.74%	Hill climbing [BIC]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	34.54%	23.80%	**41.67%**	76.20%	CTPC
Learning time (s)	Hill climbing [BIC]	37.04%	**62.96%**	0.00%	37.04%	Hill climbing [BDe]
	Hill climbing [BIC]	37.96%	**62.04%**	0.00%	37.96%	Tabu search [BIC]
	CTPC	**99.54%**	0.46%	0.00%	99.54%	Hill climbing [BIC]
	MB-CTPC	**98.15%**	1.85%	0.00%	98.15%	CTPC
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	36.11%	**63.89%**	0.00%	36.11%	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	**90.28%**	9.72%	0.00%	90.28%	Hill climbing [BIC]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	4.63%	**95.37%**	0.00%	4.63%	CTPC
Classification time (s)	Hill climbing [BIC]	31.33%	**68.67%**	0.00%	31.33%	Hill climbing [BDe]
	Hill climbing [BIC]	34.41%	**65.59%**	0.00%	34.41%	Tabu search [BIC]
	CTPC	**58.64%**	41.36%	0.00%	58.64%	Hill climbing [BIC]
	MB-CTPC	**80.25%**	19.75%	0.00%	80.25%	CTPC
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	44.75%	**55.25%**	0.00%	44.75%	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	**70.99%**	29.01%	0.00%	70.99%	Hill climbing [BIC]
	Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	**57.41%**	42.59%	0.00%	57.41%	CTPC

Table 3
Estimated performance measures (mean ± std. deviation) over the synthetic datasets.

Algorithm	Global accuracy	Mean accuracy	Macro F_1 score	Global Brier score	Learning time (s)	Classification time (s)
Hill climbing [BIC]	0.6287 ± 0.2691	0.8681 ± 0.1156	0.7753 ± 0.1971	0.4612 ± 0.3068	108.1529 ± 138.1886	6.6264 ± 9.8329
Tabu search [BIC]	0.6287 ± 0.2691	0.8681 ± 0.1156	0.7752 ± 0.1971	0.4612 ± 0.3068	107.2197 ± 137.5624	6.5689 ± 9.7887
Hill climbing [BDe]	0.6229 ± 0.2702	0.8655 ± 0.1163	0.7711 ± 0.1980	0.4678 ± 0.3076	106.4439 ± 136.8045	6.4251 ± 9.5402
CTPC	**0.6351** ± **0.2704**	**0.8729** ± **0.1125**	**0.7821** ± **0.1925**	**0.4537** ± **0.3084**	41.8573 ± 47.3675	7.5145 ± 10.5733
MB-CTPC	0.6132 ± 0.2601	0.8645 ± 0.1102	0.7698 ± 0.1889	0.4819 ± 0.2936	**21.6225** ± **22.8988**	**5.9090** ± **8.4135**
Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	0.6211 ± 0.2725	0.8658 ± 0.1165	0.7696 ± 0.1999	0.4689 ± 0.3091	49.7194 ± 57.5774	6.4525 ± 9.6591
Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]	0.6210 ± 0.2724	0.8658 ± 0.1165	0.7695 ± 0.1999	0.4691 ± 0.3090	45.8935 ± 51.4961	6.4518 ± 9.6472

5.2.2. CTPC and hill climbing

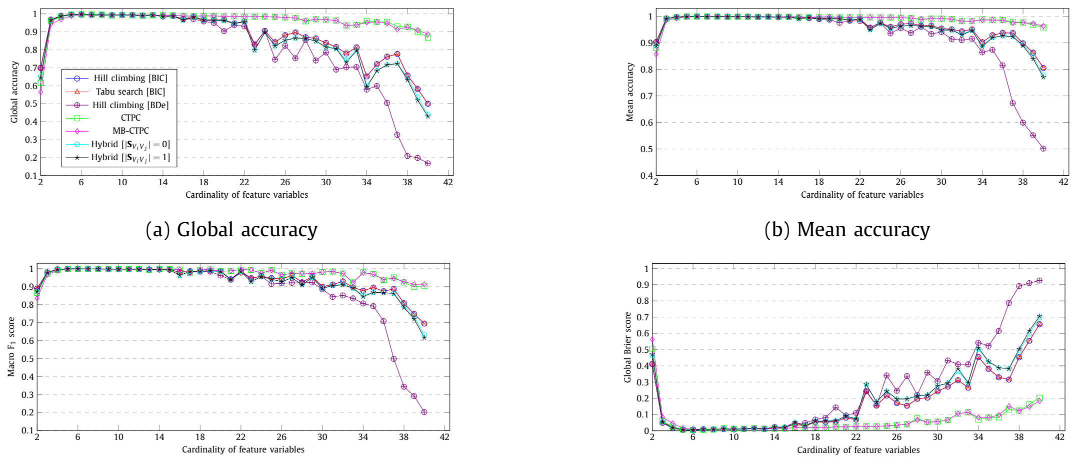
Although the mean results of Table 3 suggest that CTPC may outperform hill climbing, statistical tests show that hill climbing significantly improves all performance measures, except learning time. This was the case independently of the score optimised by hill climbing. Yet, this is the case when evaluating the algorithms on all datasets. If the datasets are divided according to the cardinality of the feature variables, hill climbing obtains better results only when the cardinality is relatively low (two, three or four states). Statistically significant improvements are made with CTPC for all performance measures when feature variables have eight possible states.



(a) Global accuracy (b) Mean accuracy

(c) Macro F$_1$ score (d) Global Brier score

Fig. 2. Results of increasing feature variables' cardinality with sequences of 10 time units.



(a) Global accuracy (b) Mean accuracy

(c) Macro F$_1$ score (d) Global Brier score

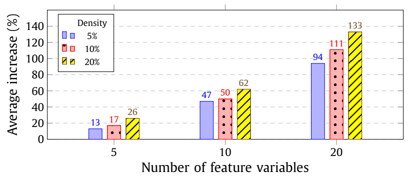
Fig. 3. Results of increasing feature variables' cardinality with sequences of 20 time units.

To further study the influence of feature variables' cardinalities, we have analysed the results of the performance measures when the cardinality is increased from two to 30. Five datasets have been sampled for each possible cardinality (145 datasets) from a single randomly generated structure with ten feature variables, four class variables, and bridge and feature subgraph densities of 10%. Fig. 2 shows that classifiers learned with constraint-based algorithms (CTPC and MB-CTPC) are more robust than score-based and hybrid solutions as feature cardinality increases. The number of examples for each possible state transition declines as the cardinality of feature variables increases, making models learned with score-based algorithms less accurate. Fig. 3 shows the results of the last experiment but using sequences with twice the duration. Increasing the sequence duration enables score-based algorithms to achieve better results with feature variables of a higher cardinality. Nevertheless, they still show worse robustness than constraint-based solutions for all the studied measures. At first glance, we thought that the BIC penalisation negatively influenced the models' accuracy. However, this behaviour is even more severe for the BDe score. We can then conclude that for problems where feature variables have high cardinality and the sequence duration is relatively small, constraint-based algorithms might be more convenient due to their robustness. This is consistent with the findings of Scutari et al. [23] and Bregoli et al. [6] for BNs and CTBNs, respectively.

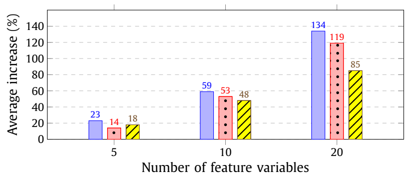
Finally, it is worth noting that constraint-based algorithms achieve much shorter learning times since the estimated parameters can be cached and quickly retrieved for future statistical tests. The usefulness of a cache is more limited for scored-based algorithms as they iteratively evaluate previously unseen parent set configurations.

5.2.3. MB-CTPC and CTPC

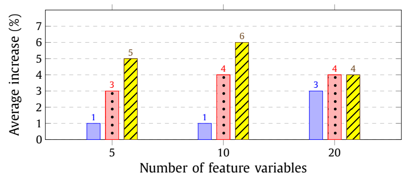
The MB-CTPC algorithm, compared to CTPC, achieves the same or, in a few cases, better results in about 60% of the datasets for global and mean accuracy and macro F$_1$ score. Simultaneously, it reduces the learning time on most datasets (98%), which was its main objective. Figs. 4a and 4b show that time differences between the algorithms are very significant,



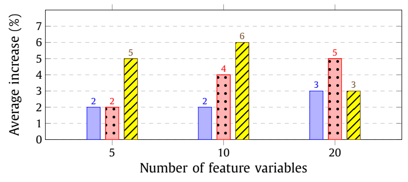
(a) Learning time by feature subgraph density



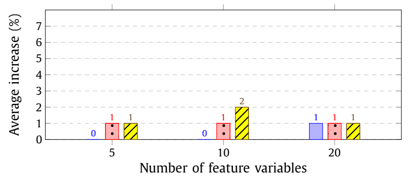
(b) Learning time by bridge subgraph density



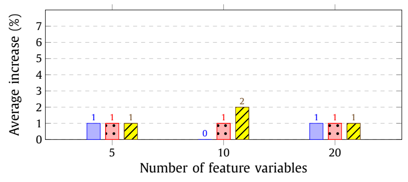
(c) Global accuracy by feature subgraph density



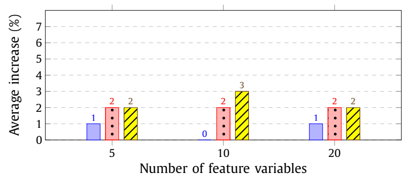
(d) Global accuracy by bridge subgraph density



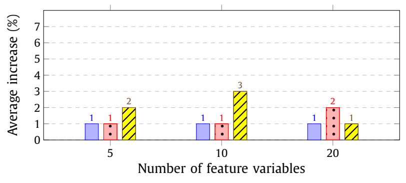
(e) Mean accuracy by feature subgraph density



(f) Mean accuracy by bridge subgraph density



(g) Macro F_1 score by feature subgraph density



(h) Macro F_1 score by bridge subgraph density

Fig. 4. Average increase in learning time, global and mean accuracy and macro F_1 score of CTPC over MB-CTPC.

Table 4
Estimated performance measures (mean ± std. deviation) over synthetic datasets generated from Multi-CTBNCs with 30 and 5 feature and class variables, respectively.

Algorithm	Global accuracy	Mean accuracy	Macro F_1 score	Global Brier score	Learning time (s)	Classification time (s)
CTPC	**0.9874** ± **0.0149**	**0.9975** ± **0.0030**	**0.9597** ± **0.0851**	**0.0238** ± **0.0286**	340.2399 ± 14.1575	217.4570 ± 37.9228
MB-CTPC	0.9590 ± 0.0403	0.9917 ± 0.0081	0.9546 ± 0.0792	0.0675 ± 0.0665	**147.0353** ± **22.6594**	**150.1994** ± **29.0154**

as the mean time increase of CTPC can reach up to 134%. The differences become more profound as the number of feature variables and feature subgraph density increase, as CTPC is more likely to perform an even higher number of unnecessary tests compared to MB-CTPC. The bridge subgraph density also influences the results, as lower density implies more significant differences for higher dimensionality data, i.e., datasets with more feature variables. The reason for this is a decrease in the number of feature variables with class variables as ancestors, reducing the tests performed by MB-CTPC. Five datasets of higher dimensionality were sampled from randomly generated structures with bridge and feature subgraph densities of 10%. Due to memory limitations, 30 feature and five class variables with six and three states, respectively, were used. Table 4 shows that MB-CTPC drastically decreases the learning time, being 57% faster than CTPC, while the classification capabilities suffer relatively negligible differences.

The CTPC algorithm obtains better results than MB-CTPC on multiple datasets, significantly improving all performance measures except learning and classification time. Nevertheless, these differences may not be significant enough if our priority is to speed up the model learning. Figs. 4c and 4d show that the mean improvement of the global accuracy can go from as little as 1% to a maximum of 6% in the performed experiments. These differences are even lower for the mean accuracy (see Figs. 4e and 4f) and macro F_1 score (see Figs. 4g and 4h). As for the global Brier score, differences in the percentages are more significant; however, they are less than 0.01 in 50% of the datasets. The slightly lower accuracy of classifiers learned with MB-CTPC arises from the incorrect definition of some class variable descendants (Step 3 of Algorithm 3). Possible causes behind this may include weak relationships between variables, training datasets not sufficiently representative of the underlying problem or the assumption that waiting times of feature variables conditioned on a non-parent ancestor follow an exponential distribution.

Overall, the classification time is also reduced when using the MB-CTPC algorithm. Table 4 shows a 31% reduction in the classification time of CTPC in the previous experiment, while a significant improvement is also seen in Tables 2 and 3. Alongside the learning time, this characteristic could make MB-CTPC more convenient for a streaming environment where models need to be incrementally updated and real-time response may be required. However, this result has to be interpreted with caution since, on certain occasions, the classification time improvement may benefit from the spouses of class variables not being defined correctly.

We can conclude that the MB-CTPC algorithm is a good choice when execution time is a priority, especially when dealing with high-dimensionality datasets. We also believe this kind of structure learning algorithm could be extended to real-time scenarios where our models have to be dynamically updated, and solutions should be provided as quickly as possible. Nevertheless, we should also consider that a trade-off exists between assuring better accuracy or significantly reducing the learning and classification times, which has to be assessed depending on the particular problem.

5.2.4. *Hybrid* $[|\mathbf{S}_{V_iV_j}| = 0]$ *and hybrid* $[|\mathbf{S}_{V_iV_j}| = 1]$

Overall, varying the maximum separating set size of the hybrid algorithm from zero to one results in no change in most experiments except for execution time. For example, the global accuracy improves in just 1.76% of the datasets when only testing for unconditional independence. However, 95% of these latter datasets have in common the presence of binary feature variables. This outcome is coherent, given that the constraint-based algorithm, which is less accurate than the score-based approaches when feature variables are binary, has more influence on the solution as the maximum size of the separating set increases. For this reason, statistically significant improvements were obtained for most performance measures with the hybrid $[|\mathbf{S}_{V_iV_j}| = 0]$ algorithm. The hybrid $[|\mathbf{S}_{V_iV_j}| = 1]$ solution succeeded in reducing the learning and classification times significantly. Nevertheless, differences between using both parameter values are generally negligible in the performed experiments.

5.2.5. *Hybrid* $[|\mathbf{S}_{V_iV_j}| = 0]$ *vs. hill climbing [BIC] and CTPC*

Considering all synthetic datasets, no significant differences were found in terms of classification capabilities between the hybrid $[|\mathbf{S}_{V_iV_j}| = 0]$ and CTPC algorithms, but an improvement in global Brier score and classification time by the former and in learning time by the latter. When comparing the hybrid $[|\mathbf{S}_{V_iV_j}| = 0]$ and hill climbing [BIC] algorithms, the classification capabilities of the former are significantly diminished, yet the learning and classification times are substantially improved.

Considering only datasets with low cardinality feature variables (two, three and four states), the statistical tests report an improvement in all performance measures, except execution times, by the hybrid algorithm with respect to CTPC while still notably improving the learning and, to a lesser extent, classification times of hill climbing [BIC]. Thus, the proposed algorithm achieves its intended purpose as an intermediate solution, as it combines the better classification capabilities of score-based solutions with the faster learning time of constraint-based approaches in the case of dealing with low cardinality feature variables. As discussed in Section 5.2.2, the CTPC algorithm performs significantly worse than the score-based approaches when dealing with low cardinality feature variables. Then, combining these two approaches in a hybrid algorithm improves the CTPC performance.

5.2.6. *Robustness to noisy data*

In this section, we evaluate the robustness of each structure learning algorithm by progressively increasing the presence of noise in the data. To do so, ten Multi-CTBNCs with 20 feature variables, four class variables and bridge and feature subgraph densities of 10% were randomly defined. Four datasets were sampled from each of these models with the following degrees of noise:

- Noise-free dataset: no noise was added when sampling this dataset.
- Low noise dataset: 5% of feature variables' transitions and class variables' states were randomly sampled, while a Gaussian noise with zero mean and standard deviation 0.1 was added to feature variables' transition times.
- Medium noise dataset: 10% of feature variables' transitions and class variables' states were randomly sampled, while a Gaussian noise with zero mean and standard deviation 0.2 was added to feature variables' transition times.
- High noise dataset: 20% of feature variables' transitions and class variables' states were randomly sampled, while a Gaussian noise with zero mean and standard deviation 0.5 was added to feature variables' transition times.

Overall, most algorithms were similarly affected by the increased noise in the datasets. However, this was different for the MB-CTPC algorithm. Table 5 compares the results of the CTPC and MB-CTPC algorithms for the given noise levels and shows a greater difference between them as the noise increases. As we have discussed in Section 5.2.3, the definition of the class variables' descendants is a crucial step for the correct functioning of the algorithm, which is being increasingly affected by adding more noise. On the other hand, the CTPC algorithm is more robust, as it simply tests the dependencies between all possible pairs of feature variables. Nevertheless, this translates into a much higher cost in the learning time, which is almost twice that of the MB-CTPC algorithm, and in the classification time.

Table 5
Estimated performance measures (mean ± std. deviation) when increasing noise in synthetic datasets.

Dataset	Global accuracy	Mean accuracy	Macro F_1 score	Global Brier score	Learning time (s)	Classification time (s)
CTPC						
Noise-free dataset	0.7927 ± 0.2217	0.9458 ± 0.0595	0.9202 ± 0.0878	0.2797 ± 0.2926	102.5735 ± 4.0715	25.0581 ± 2.9236
Low noise dataset	0.7898 ± 0.2228	0.9443 ± 0.0618	0.9198 ± 0.0899	0.2891 ± 0.2977	93.1212 ± 4.0769	22.5176 ± 2.6766
Medium noise dataset	0.7781 ± 0.2171	0.9411 ± 0.0605	0.9163 ± 0.0879	0.3028 ± 0.2857	76.6488 ± 3.8178	19.1597 ± 2.1732
High noise dataset	0.7151 ± 0.1841	0.9198 ± 0.0558	0.8885 ± 0.0796	0.3882 ± 0.2350	52.2272 ± 2.4204	13.0160 ± 1.3833
MB-CTPC						
Noise-free dataset	0.7743 ± 0.2057	0.9412 ± 0.0556	0.9151 ± 0.0833	0.3065 ± 0.2692	54.2652 ± 6.0504	19.1903 ± 1.9290
Low noise dataset	0.7611 ± 0.2075	0.9360 ± 0.0591	0.9107 ± 0.0851	0.3284 ± 0.2737	47.5664 ± 3.8766	17.1980 ± 1.8136
Medium noise dataset	0.7234 ± 0.2000	0.9244 ± 0.0585	0.8982 ± 0.0811	0.3755 ± 0.2601	38.0962 ± 2.9495	14.2572 ± 1.4947
High noise dataset	0.6477 ± 0.1960	0.8933 ± 0.0696	0.8595 ± 0.0836	0.4655 ± 0.2388	25.0125 ± 1.5644	9.8222 ± 0.9120

5.3. Experimental results on a real-world dataset

This section evaluates the effectiveness of the introduced structure learning algorithms when solving a real-world problem. For this purpose, the British household panel survey (BHPS) dataset has been used, which is the result of a longitudinal study gathering, among others, information about the finances, health, household, work life, opinions and other personal aspects of UK citizens [30]. The surveys were conducted annually from 1991 to 2009 in a total of 18 waves, all of which are used in this work, except for the first, fourth and ninth waves, as some variables relevant to this study were not collected.

The objective of this experiment is to predict the state of certain variables related to personal information, given the evolution of individuals' responses over the years. The usefulness of this work lies in the possibility of extracting relationships from a particular dataset and applying them to infer new knowledge in, for example, other surveys in which specific information was not collected. The following seven class variables were defined for this experiment:

- Dental check-up: denotes whether the individual has had a dental check-up in the last year.
- Employment status: specifies the individual's current employment status. This variable takes ten states: self-employed, in paid employment, unemployed, retired, maternity leave, looking after family or home, full-time student, long-term sick or disabled, on a government training scheme or others.
- Limb, back or neck problems: whether the respondent has problems or disabilities related to the arms, legs, hands, feet, back or neck (including arthritis and rheumatism).
- Lives with spouse or partner: whether the respondent is living with their spouse or partner.
- Responsible adult for child: whether the individual is responsible for a child under 16 years old.
- Sex: sex of the individual.
- Smoker: specifies if the individual smokes.

The BHPS dataset collects information from 29702 individuals on more than 1300 variables. For the definition of the feature variables, we focused on a subset of discrete-state variables related to health, work life, household and other personal information, such as marital status or residence region. Due to the nature of these data, the BHPS dataset is largely incomplete, as respondents could be unable or refuse to answer certain questions. Therefore, we have kept those variables containing no more than 3% of missing data. Altogether, 26 feature variables of different kinds have been used (see Fig. 5).

Each sequence extracted from the BHPS dataset contains the survey results of a single individual, results that are ordered according to the date they were collected and taking into account that the state of the class variables (class configuration) does not vary along a sequence. Therefore, as many sequences have been extracted for each individual as changes in the class configuration plus one. Sequences containing only one observation were discarded since they do not provide any information. This may occur, for example, if an individual took the survey on a single occasion. In total, 14925 sequences were extracted.

Ten cross-validations with random shuffles have been performed to statistically compare the performance of the structure learning algorithms on the BHPS dataset. The average results of this experiment are shown in Table 6. Despite obtaining better results in more synthetic experiments with score-based algorithms (see Fig. 2 and Section 5.2.2), a substantial improvement in classification performance was achieved by constraint-based algorithms on this occasion. In addition, hill climbing results were significantly improved by optimising the BDe score instead of BIC, which was not the case with the synthetic datasets. These circumstances illustrate the importance of exploring structure learning algorithms of different natures, as they could obtain better results in diverse settings. As for the differences between the CTPC and MB-CTPC algorithms, the proposed method significantly improved all performance measures. The MB-CTPC algorithm reduced the learning time of CTPC by more than 50% while delivering a similar classification performance. Finally, the hybrid algorithm exhibited the worst overall results for this experiment, except for the classification time when using a maximum separating set size of two. If the size is reduced, the results worsen for this experiment and the computational time increases. In the case of an empty separating set, the available computational resources were not enough to learn the structures. This is because

Table 6
Estimated performance measures (mean ± std. deviation) with ten cross-validations on the British household panel survey dataset.

Algorithm	Global accuracy	Mean accuracy	Macro F_1 score	Micro F_1 score	Global Brier score	Learning time (s)	Classification time (s)
Hill climbing [BIC]	0.2029 ± 0.0014	0.7772 ± 0.0009	0.5614 ± 0.0013	0.7536 ± 0.0011	0.8991 ± 0.0004	11.4602 ± 0.5081	3.7173 ± 0.0877
Tabu search [BIC]	0.2029 ± 0.0014	0.7772 ± 0.0009	0.5614 ± 0.0013	0.7536 ± 0.0011	0.8991 ± 0.0004	10.9979 ± 0.3562	3.7071 ± 0.1446
Hill climbing [BDe]	0.2184 ± 0.0023	0.7902 ± 0.0008	0.5668 ± 0.0026	0.7668 ± 0.0009	0.8855 ± 0.0017	**10.8260 ± 0.2263**	3.1707 ± 0.0616
CTPC	0.3799 ± 0.0020	0.8601 ± 0.0009	0.6608 ± 0.0022	0.8439 ± 0.0009	0.7931 ± 0.0014	47.5145 ± 7.7559	5.5352 ± 0.0872
MB-CTPC	**0.3849 ± 0.0012**	**0.8614 ± 0.0005**	**0.6638 ± 0.0015**	**0.8445 ± 0.0005**	**0.7866 ± 0.0008**	23.4605 ± 3.0436	5.2359 ± 0.0718
Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 0$]	-	-	-	-	-	-	-
Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 1$]	0.1573 ± 0.0021	0.7653 ± 0.0007	0.4898 ± 0.0033	0.7303 ± 0.0025	0.9212 ± 0.0003	57.7243 ± 17.4723	3.4126 ± 0.0552
Hybrid [$\lvert\mathbf{S}_{V_iV_j}\rvert = 2$]	0.1597 ± 0.0022	0.7691 ± 0.0007	0.5072 ± 0.0031	0.7338 ± 0.0023	0.9194 ± 0.0003	17.8435 ± 0.5907	**3.0866 ± 0.0769**

the hybrid algorithm's restriction phase can only remove a few arcs, reporting a very dense structure to the maximisation phase. This causes estimating a large set of parameters for nodes with multiple parents.

Fig. 5 shows a structure learned with the MB-CTPC algorithm. Although an exhaustive analysis of the variables' dependencies is beyond the scope of this work, some meaningful relationships can be easily identified thanks to the graphical capabilities of the model:

- The class variable *Limb, back or neck problems* is directly related to other medical variables, including those indicating limitations in daily and work activities for medical reasons; for example, individuals who claim to have these health problems are more likely to report that their health is limiting their work in the upcoming years.
- The individuals' employment status was found to be related to their health condition, type of household or if they are of working age; for example, given that an individual is retired but still of working age (pre-retired), they are expected to be above working age earlier than if they had any other employment status. Specifically, they are expected to do so in about 4 to 5 years.
- The evolution of an individual's household type is naturally affected by variables such as the number of people living in the household or whether the individual lives with a spouse or partner. However, this variable is also affected by other factors such as employment status; for example, given an employed person living with a partner in a two-person household, the likelihood of the household type changing from couple without children to couple with dependent children from one survey to the following is greater than if, for example, the individual is retired.
- Given an individual whose health limits their work, it is more likely that their health will also affect their daily tasks in a shorter time than if they did not have difficulties at work. The same is true when changing the occurrence order.
- The evolution of the household type provides information on whether an individual is living with a spouse or partner over a period of time; for example, it is evident that a household transition from single-parent with dependent children to single-parent with non-dependent children only occurs for individuals who are not living with a spouse or partner.

The interest in using a Multi-CTBNC to model the BHPS dataset lies in the fact that dependencies also exist between the class variables. As an example, the following relationships have been found:

- Given that an individual has had a dental check-up in the last year, they are likelier to be a non-smoker.
- Individuals are more likely to be responsible for a child when they had a dental check-up last year. We could hypothesise that this is due to the individual being more responsible when becoming a parent and the intention of being a healthy role model for their children.
- The probability that an individual is responsible for a child is higher if they are employed or self-employed than unemployed. Moreover, this probability is significantly increased if the individual has indicated that their employment status is family or home care. On the opposite side, the probability of being responsible for a child is very low if the individual is retired or a student.

6. Conclusions and future work

This article introduces, for the first time, constraint-based and hybrid structure learning algorithms for continuous-time Bayesian network classifiers, which were specially designed to learn Multi-CTBNCs. The novel constraint-based algorithm, named MB-CTPC, aims to learn the structure of these classifiers by performing conditional independence tests only on dependencies that could be relevant to the Markov blankets of class variables. Then, the hybrid algorithm, a solution not even studied for CTBNs, seeks to combine the strengths of the score- and constraint-based methods.

Synthetic experiments show that the MB-CTPC algorithm significantly improves the learning and classification times of Multi-CTBNCs compared to other structure learning algorithms while maintaining a competitive classification performance. This algorithm is especially convenient when learning from high-dimensionality datasets, but significant improvements were obtained regardless of the number of variables, their cardinality or structure density. Furthermore, the proposed algorithm was statistically proven to be the best choice when used on a real-world dataset, as it matched the best results achieved by the constraint-based algorithm CTPC while significantly reducing the learning and classification time of the models. Finally,

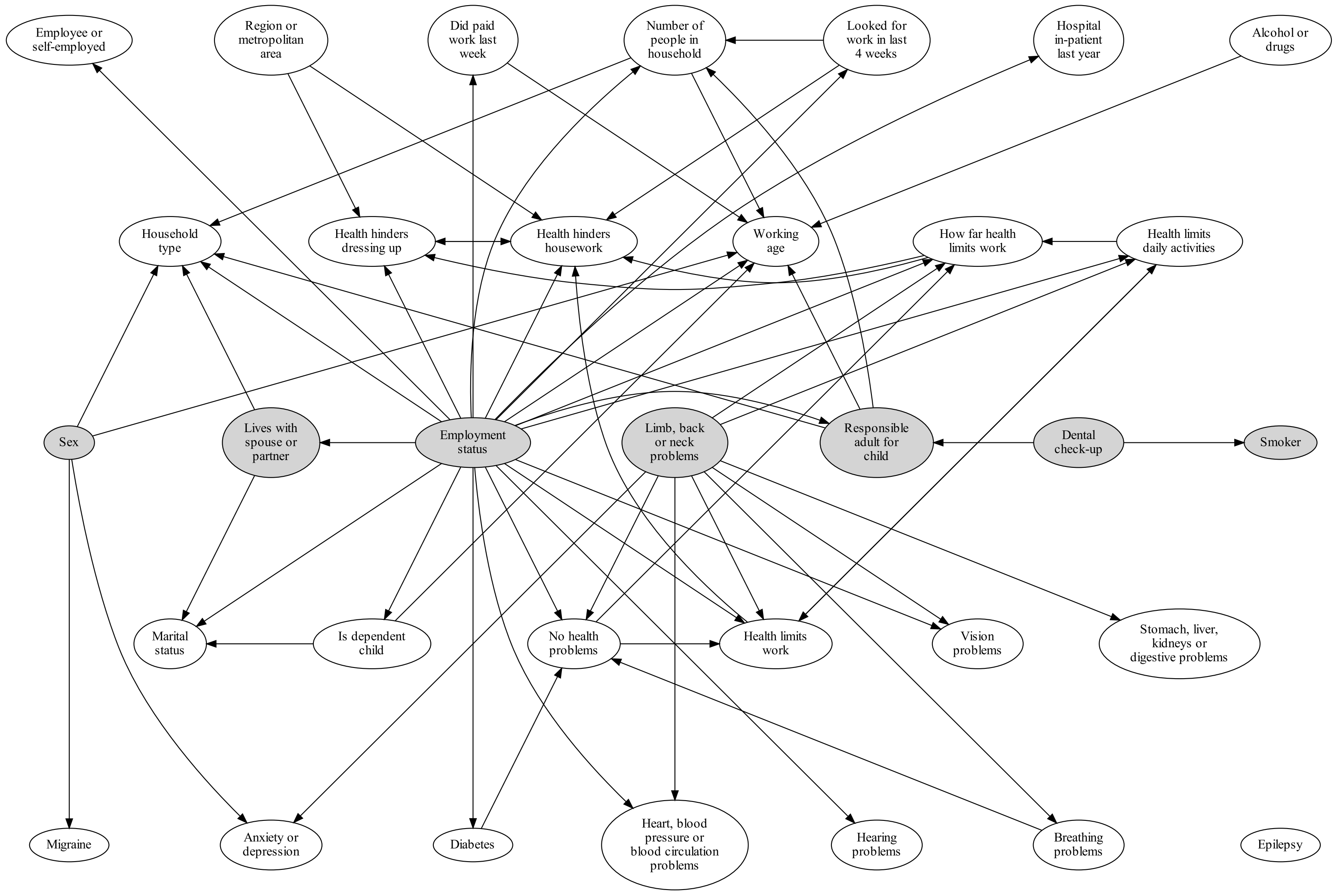

Employee or self-employed
Region or metropolitan area
Did paid work last week
Number of people in household
Looked for work in last 4 weeks
Hospital in-patient last year
Alcohol or drugs
Household type
Health hinders dressing up
Health hinders housework
Working age
How far health limits work
Health limits daily activities
Sex
Lives with spouse or partner
Employment status
Limb, back or neck problems
Responsible adult for child
Dental check-up
Smoker
Marital status
Is dependent child
No health problems
Health limits work
Vision problems
Stomach, liver, kidneys or digestive problems
Migraine
Anxiety or depression
Diabetes
Heart, blood pressure or blood circulation problems
Hearing problems
Breathing problems
Epilepsy

Fig. 5. Structure learned on the British household panel survey dataset with the MB-CTPC algorithm (class variables are highlighted with a grey background).

the hybrid algorithm provides, in certain scenarios, an intermediate solution that significantly improves the results of CTPC and drastically reduces the learning and classification times of score-based techniques.

Multiple areas of open research were found while conducting this work:

(i) MB-CTPC may struggle to identify descendants of class variables. Using a phase distribution, such as Erlang [17], to model the transition times of feature variables conditioned on ancestor class variables may improve this task and the model's accuracy. We would also like to study whether this improvement could contribute to mitigating the impact of noise.

(ii) Discerning between descendants and children of class variables may improve MB-CTPC learning time, as arcs to feature nodes with no parent class variables are irrelevant for the classification task.

(iii) When dealing with datasets of high dimensionality and size, multiple class variables and an underlying structure of considerable density, the classification phase can be computationally expensive (see, for example, Table 4). Thus, a class-bridge decomposable Multi-CTBNC [4] could be appropriate to improve this aspect.

(iv) Information from class variable relationships may be useful to improve the definition of class descendants and, therefore, the accuracy of MB-CTPC.

(v) A first hybrid algorithm was proposed to solve the structure learning problem for CTBNs. However, the improvements achieved compared to other solutions are limited to specific scenarios. Further study of these algorithms is needed to determine their usefulness in real situations, as well as to improve their results in more settings. As a starting point, attempts could be made to extend some ideas used in existing hybrid algorithms for other PGMs, such as [28] or [11].

(vi) The HITON algorithm [3] could be adapted for learning the structure of Multi-CTBNCs. This algorithm has already been successfully applied to the learning of multidimensional Bayesian network classifiers [5]. Therefore, we are interested in assessing the performance of this proposal with respect to the solutions studied in this paper.

(vii) Employing feature subset selection techniques prior to structure learning could improve execution times and classification results since irrelevant and redundant variables that may confuse learning algorithms are ignored. However, there is little literature on feature selection for categorical time series and even less on a supervised context. This approach would be useful for problems such as the one introduced with the BHPS dataset, which consists of more than 1300 feature variables.

(viii) The proposed MB-CTPC structure learning algorithm could be suitable for streaming data classification problems where models are dynamically updated, as the relationships between variables can change over time, and solutions are provided as fast as possible. Therefore, we plan to evaluate the usefulness and effectiveness of an adaptation of this algorithm for such a scenario.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The link to our data/code is included in the manuscript. We do not have permission to share the British Household Panel Survey data.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Science, Innovation and Universities through the PID2019-109247GB-I00, RTC2019-006871-7 and TED2021-131310BI00 projects. The British Household Panel Survey is produced by the Institute for Social and Economic Research, University of Essex, with the support of the Economic and Social Research Council. The research data are distributed by the UK Data Service. C. Villa-Blanco is supported by a predoctoral contract for the formation of doctors from the Universidad Politécnica de Madrid.

References

[1] E. Acerbi, F. Stella, Continuous time Bayesian networks for gene network reconstruction: a comparative study on time course data, in: International Symposium on Bioinformatics Research and Applications, Springer, 2014, pp. 176–187.

[2] S. Acid, L.M. De Campos, A hybrid methodology for learning belief networks: BENEDICT, Int. J. Approx. Reason. 27 (3) (2001) 235–262.

[3] C.F. Aliferis, I. Tsamardinos, A. Statnikov, HITON: a novel Markov blanket algorithm for optimal variable selection, in: AMIA Annual Symposium Proceedings, American Medical Informatics Association, 2003, pp. 21–25.

[4] C. Bielza, G. Li, P. Larrañaga, Multi-dimensional classification with Bayesian networks, Int. J. Approx. Reason. 52 (6) (2011) 705–727.

[5] H. Borchani, C. Bielza, P. Martínez-Martín, P. Larrañaga, Markov blanket-based approach for learning multi-dimensional Bayesian network classifiers: an application to predict the European quality of life-5 dimensions (EQ-5D) from the 39-item Parkinson's disease questionnaire (PDQ-39), J. Biomed. Inform. 45 (6) (2012) 1175–1184.

[6] A. Bregoli, M. Scutari, F. Stella, A constraint-based algorithm for the structural learning of continuous-time Bayesian networks, Int. J. Approx. Reason. 138 (2021) 105–122.

[7] D. Codecasa, F. Stella, Learning continuous time Bayesian network classifiers, Int. J. Approx. Reason. 55 (8) (2014) 1728–1746.

[8] D. Colombo, M.H. Maathuis, Order-independent constraint-based causal structure learning, J. Mach. Learn. Res. 15 (1) (2014) 3741–3782.
[9] J.A. Fernandes, J.A. Lozano, I. Inza, X. Irigoien, A. Pérez, J.D. Rodríguez, Supervised pre-processing approaches in multiple class variables classification for fish recruitment forecasting, Environ. Model. Softw. 40 (2013) 245–254.
[10] T.-C. Fu, A review on time series data mining, Eng. Appl. Artif. Intell. 24 (1) (2011) 164–181.
[11] M. Gasse, A. Aussem, H. Elghazel, A hybrid algorithm for Bayesian network structure learning with application to multi-label learning, Expert Syst. Appl. 41 (15) (2014) 6755–6772.
[12] S. Gil-Begue, C. Bielza, P. Larrañaga, Multi-dimensional Bayesian network classifiers: a survey, Artif. Intell. Rev. 54 (1) (2021) 519–559.
[13] M. Henrion, Propagating uncertainty in Bayesian networks by probabilistic logic sampling, in: Machine Intelligence and Pattern Recognition, vol. 5, 1988, pp. 149–163.
[14] W.-C. Juang, S.-J. Huang, F.-D. Huang, P.-W. Cheng, S.-R. Wann, Application of time series analysis in modelling and forecasting emergency department visits in a medical centre in Southern Taiwan, BMJ Open 7 (11) (2017) e018628.
[15] B.-H. Kim, J.-Y. Pyun, ECG identification for personal authentication using LSTM-based deep recurrent neural networks, Sensors 20 (11) (2020) 3069.
[16] H. Liu, S. Zhou, W. Lam, J. Guan, A new hybrid method for learning Bayesian networks: separation and reunion, Knowl.-Based Syst. 121 (2017) 185–197.
[17] M. Liu, F. Stella, A. Hommersom, P.J. Lucas, Making continuous time Bayesian networks more flexible, in: International Conference on Probabilistic Graphical Models, PMLR, 2018, pp. 237–248.
[18] M. Liu, F. Stella, A. Hommersom, P.J. Lucas, L. Boer, E. Bischoff, A comparison between discrete and continuous time Bayesian networks in learning from clinical time series data with irregularity, Artif. Intell. Med. 95 (2019) 104–117.
[19] Z. Meng, S. Han, Y. Tong, Listen to your face: inferring facial action units from audio channel, IEEE Trans. Affect. Comput. 10 (4) (2017) 537–551.
[20] U. Nodelman, C.R. Shelton, D. Koller, Continuous time Bayesian networks, in: Proceedings of the 18th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, 2002, pp. 378–387.
[21] U. Nodelman, C.R. Shelton, D. Koller, Learning continuous time Bayesian networks, in: Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, 2003, pp. 451–458.
[22] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, 1988.
[23] M. Scutari, C.E. Graafland, J.M. Gutiérrez, Who learns better Bayesian network structures: accuracy and speed of structure learning algorithms, Int. J. Approx. Reason. 115 (2019) 235–253.
[24] P. Spirtes, C.N. Glymour, R. Scheines, D. Heckerman, Causation, Prediction, and Search, The MIT Press, 2000.
[25] F. Stella, Y. Amer, Continuous time Bayesian network classifiers, J. Biomed. Inform. 45 (6) (2012) 1108–1119.
[26] G.A. Susto, A. Cenedese, M. Terzi, Time-series classification methods: review and applications to power systems data, in: Big Data Application in Power Systems, 2018, pp. 179–220.
[27] G. Trabelsi, P. Leray, M. Ben Ayed, A.M. Alimi, Dynamic MMHC: a local search algorithm for dynamic Bayesian network structure learning, in: International Symposium on Intelligent Data Analysis, Springer, 2013, pp. 392–403.
[28] I. Tsamardinos, L.E. Brown, C.F. Aliferis, The max-min hill-climbing Bayesian network structure learning algorithm, Mach. Learn. 65 (1) (2006) 31–78.
[29] R.S. Tsay, Analysis of Financial Time Series, John Wiley & Sons, 2005.
[30] University of Essex, Institute for Social and Economic Research, British Household Panel Survey: Waves 1-18, 1991-2009, 8th edition, 2018 [data collection], https://doi.org/10.5255/UKDA-SN-5151-2.
[31] S. Villa, F. Stella, Learning continuous time Bayesian networks in non-stationary domains, J. Artif. Intell. Res. 57 (2016) 1–37.
[32] C. Villa-Blanco, P. Larrañaga, C. Bielza, Multidimensional continuous time Bayesian network classifiers, Int. J. Intell. Syst. 36 (12) (2021) 7839–7866.
[33] J. Xu, C.R. Shelton, Intrusion detection using continuous time Bayesian networks, J. Artif. Intell. Res. 39 (2010) 745–774.