Received: 8 August 2021 | Accepted: 11 November 2021

DOI: 10.1002/int.22770

RESEARCH ARTICLE

WILEY

Multipartition clustering of mixed data with Bayesian networks

Fernando Rodriguez-Sanchez | **Concha Bielza** | **Pedro Larrañaga**

Computational Intelligence Group, Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Madrid, Spain

Correspondence
Fernando Rodriguez-Sanchez, Computational Intelligence Group, Departamento de Inteligencia Artificial, Universidad Politécnica de Madrid, Madrid, Spain.
Email: fernando.rodriguezs@upm.es

Funding information
Ministerio de Economía y Competitividad, Grant/Award Number: PID2019-109247GB-I00; Horizon 2020, Grant/Award Number: 945539; Fundación BBVA, Grant/Award Number: Score-based nonstationary temporal Bayesian networks. Applications in climate and neuroscience

Abstract

Real-world applications often involve multifaceted data with several reasonable interpretations. To cluster this data, we need methods that are able to produce multiple clustering solutions. To this purpose, it is interesting to learn a finite mixture model with multiple latent variables, where each latent variable represents a unique way to partition the data. However, although there is an extensive literature on multipartition clustering methods for categorical data and for continuous data, there is a lack of work for mixed data. In this paper, we propose a multipartition clustering method that is able to efficiently deal with mixed data by exploiting the Bayesian network factorization and the variational Bayes framework. We show the flexibility and applicability of the proposed method by solving clustering, density estimation, and missing data imputation tasks in real-world data sets. For reproducibility, all code, data, and results can be found in the following public repository: https://github.com/ferjorosa/mpc-mixed.

KEYWORDS

Bayesian networks, mixed data, model-based clustering, multipartition clustering

1 | INTRODUCTION

Traditional model-based clustering methods assume the existence of a finite mixture model[1] that partitions the data in a single way. However, this assumption is often untrue for real-world data sets, where several meaningful sets of clusters may exist, each of them associated to a specific subset of data variables. For instance, one can imagine that different subsets of disease symptoms (e.g., muscular disorders, psychological issues, organ dysfunctions, etc.) that describe a cohort of patients can provide different ways of grouping those patients. In this setting, multipartition clustering extends model-based clustering by assuming the existence of a finite mixture model that partitions the data in multiple ways, where each partition is defined by a different subset of the domain variables.

The problem of multipartition clustering was first addressed by Zhang[2] for categorical data and by Galimberti and Soffritti[3] for continuous data. Since then, several categorical[4-9] and continuous[10-14] multipartition clustering methods have been proposed. However, there has been relatively little attention to the problem of finding multiple partitions of mixed data.[15,16] One of the main reasons for this lack of methods is the added difficulty of representing and learning mixture models when dealing with mixed data.[17] As such, our main concern in this paper is to propose a multipartition clustering method that is able to effectively deal with mixed data. To this end, we will exploit the conditional independences that are present in the data using a Bayesian network (BN).[18,19]

1.1 | Multipartition clustering with Bayesian networks

Let $\mathcal{D}$ be a data set with a set of N independent, identically distributed data instances with an associated set of categorical and continuous observed variables $\mathbf{X} = \{X_1, ..., X_m\}$. Multipartition clustering algorithms assume that data has been generated by a probability distribution $P(\mathbf{X})$ that can be expressed as a finite mixture model with s categorical latent variables $\mathbf{H} = \{H_1, ..., H_s\}$. Each latent variable in the model represents a unique partition (i.e., clustering solution) with k_i probabilistic clusters:

$$P(\mathbf{X}) = \sum_{H_1} ... \sum_{H_s} P(\mathbf{H}) P(\mathbf{X}|\mathbf{H}). \tag{1}$$

However, working with this model becomes cumbersome because the number of model parameters increases exponentially with respect to the number of categorical variables. Furthermore, its interpretation also becomes difficult because each partition is defined by all the observed variables, regardless of their relevance to it. It is therefore necessary to exploit the conditional independences that are present in the data. Conditional independence is a central concept to BNs. When conditional independences are present, BNs produce a factorization of the joint probability distribution that substantially reduces the number of model parameters and allows to graphically represent relevant relationships between variables.

Given a set of variables $\mathbf{Y} = \{\mathbf{X}, \mathbf{H}\}$, a BN is defined by: (i) a *directed acyclic graph* $\mathcal{G}$ that comprises the structure of the network and represents the conditional independences among the variables, and (ii) a set of *parameters* θ that represents the conditional probability distribution (CPD) of each variable $Y_i \in \mathbf{Y}$ given its parents $\mathbf{Pa}_i^{\mathcal{G}}$ in the graph. $\mathcal{B} = \{\mathcal{G}, \theta\}$ is a BN

with respect to $\mathcal{G}$ if and only if it satisfies the local Markov property, that is, each variable is conditionally independent of its nondescendants given its parents in the graph. Hence, the joint probability distribution factorizes as follows:

$$P(\mathbf{Y}) = \prod_i P\left(Y_i | \mathbf{Pa}_i^{\mathcal{G}}\right). \tag{2}$$

Depending on the nature of the variables that are present in a BN, we can distinguish between categorical, continuous and mixed BNs. In this paper, we are interested in clustering mixed data. Therefore, we are going to work with mixed BNs, which are composed of categorical and continuous variables. More specifically, we are going to consider conditional linear Gaussian (CLG) BNs.[20] In a CLG BN, every categorical variable may only have categorical parents and its CPD is categorical. In addition, every continuous variable may have both categorical and continuous parents, and its CPD is a CLG. Let Y be a continuous variable where $\mathbf{C}$ are its continuous parents and $\mathbf{D}$ are its categorical parents. We say that Y follows a CLG distribution if for every assignment $\mathbf{d} \in \Omega_{\mathbf{D}}$ (where $\Omega_{\mathbf{D}}$ represents all possible joint assignments to $\mathbf{D}$) there are Gaussian parameters $\beta_{\mathbf{d},i}$ and $\sigma^2_{\mathbf{d}}$, such that

$$P(Y|\mathbf{C}, \mathbf{d}) = \mathcal{N}\left(\beta_{\mathbf{d},0} + \sum_i \beta_{\mathbf{d},i} C_i;\ \sigma^2_{\mathbf{d}}\right). \tag{3}$$

Learning a BN from data is typically performed by a search method that approaches the learning process from a model selection perspective. Search methods define a hypothesis space of potential models, a set of operators to navigate this space, and a scoring function that measures how well the model fits the observed data.[21] When all the variables in the network are observed, the decomposability property of BN scores such as the Akaike information criterion (AIC),[22] and the Bayesian information criterion (BIC)[23] allows for efficient learning. However, in the presence of latent variables it is not feasible to efficiently learn a BN because scoring functions do not decompose.

Friedman's structural expectation–maximization (SEM)[24] is the most widely used algorithm for learning a BN in the presence of latent variables. It generalizes the expectation–maximization (EM) algorithm[25] to the problem of BN learning. In the expectation step, it uses the current model to estimate the values of latent variables and generates a completed data set $\mathcal{D}^*$. Then, in the maximization step, it estimates the parameters and structure of the new model using the completed data. Any search method can be used in the maximization step. However, the scoring function to be maximized must be a penalized version of the log-likelihood (LL), such as AIC or BIC.

It is important to note the benefits of SEM compared to a brute-force approach: rather than re-estimating the model parameters after each structure change, the output of a single expectation step is used to perform many structure changes. At each iteration t, SEM selects the model $\mathcal{B}_{t+1}$ with the highest expected score. The expected score is computed over the completed data and it is referred to as $\text{score}(\mathcal{B}_{t+1} : \mathcal{D}_t^*)$. Its use is motivated by the following inequality:

$$\text{score}\left(\mathcal{B}_{t+1} : \mathcal{D}_t^*\right) - \text{score}\left(\mathcal{B}_t : \mathcal{D}_t^*\right) \leq \text{score}(\mathcal{B}_{t+1} : \mathcal{D}) - \text{score}(\mathcal{B}_t : \mathcal{D}), \tag{4}$$

which states that a score improvement with respect to the completed data guarantees an improvement with respect to the observed data. Hence, Equation (4) ensures that the SEM algorithm converges to a local optimum without the need of using probabilistic inference in each structure change.

When a latent variable is known to exist, we can introduce it into the BN model and apply SEM. However, when performing multipartition clustering, we do not know either the appropriate number of latent variables, nor their respective cardinalities. Current literature[2-12] has approached this problem by constructing the BN model one local move at a time (i.e., introducing a latent variable, removing an arc, increasing the cardinality of a latent variable, etc.). Surprisingly, not much attention has been paid to incorporate the SEM algorithm into the learning process. As a result, existing approaches limit their search of models to tree-like structures to reduce their computational complexities. Finally, another limitation of current methods is their inability to learn from mixed data.

1.2 | Our proposal

In this paper, we develop a multipartition clustering method that incorporates a variational Bayesian (VB)[26] version of SEM to learn CLG BNs with categorical latent variables from mixed data. Therefore, the main contributions of our work are as follows:

- We present the VB-SEM algorithm.
- We propose a greedy search method for learning CLG BNs with categorical latent variables. This method searches the space of models using five latent operators and a local version of VB-SEM.

Our empirical results show that our proposed method is able to reveal interesting partitions when applied to real-world data. In addition, it achieves state-of-the-art performance in density estimation tasks with mixed data.

1.3 | Other related work

Besides multipartition clustering, there are other lines of work that generate multiple clustering solutions without using mixture models with multiple latent variables. They are usually referred to as multiple clustering methods and can be divided into two groups: unsupervised multiple clustering methods and semisupervised multiple clustering methods. The first group of algorithms identify multiple clustering solutions without taking into consideration any previously identified ones. To this end, they may use: (i) different initializations,[27] (ii) nonparametric Bayesian models,[28] (iii) orthogonal subspaces,[29] and (iv) independence subspace analysis.[30] In contrast, the second group of algorithms identify multiple clustering solutions that are distinct to the previously identified ones. They often consider: (i) minimizing the mutual information between clustering solutions,[31] (ii) using nonnegative matrix factorization,[32] and (iii) maximizing the distance between clustering solutions.[33]

Multipartition and multiple clustering methods should not be confused with those of multiview and ensemble clustering, which aim at producing a single unified clustering solution.[34] Multiview clustering methods search for clusterings in different subspaces and then

combine them into a single solution.[35, 36] Ensemble clustering methods create a series of diverse base clusterings (i.e., by using different subsets of variables, different subsets of data instances or different clustering algorithms) and then combine them to produce a unified solution.[37]

Finally, multipartition clustering is generally associated to the problem of unsupervised feature selection.[10] Multipartition clustering intrinsically performs feature selection when defining the partitions. Each partition is formed by a subset of related variables. Those variables that are not relevant in any partition end up being independent. This process is similar to the proposal of Tang et al.,[38] which develops an unsupervised feature selection algorithm for multiview data, in which variables from the same view and from different views are related using a cross-view similarity graph.

1.4 | Organization of the paper

The remainder of this paper is organized as follows. In Section 2, we describe the VB-SEM algorithm. In Section 3, we present our greedy search method for learning CLG BNs with categorical latent variables. In Section 4, we evaluate the performance of our method in a clustering task with Parkinson's data, in a density estimation task with several real-world mixed data sets, and in a missing data imputation task with real-world data. Finally, we present our conclusions in Section 5. Additionally, Table 1 provides a list of the abbreviations and acronyms used in this paper.

2 | VB-SEM

In this section, we introduce a variant of SEM that it is framed within the VB framework to ensure its applicability to mixed data. We refer to this method as VB-SEM. The introduction of the VB framework arises from the need to apply probabilistic inference in the expectation step of SEM. Since exact inference poses a space complexity problem (i.e., even representing the correct marginal distribution in a CLG BN may require space that is exponential in the size of network[39]), approximate inference is required. Two of the most popular approximation schemes are Markov chain Monte Carlo (MCMC)[40] and variational inference (VI).[41] We chose VI (and more specifically the VB framework) due to its speed advantage over MCMC and its easy integration with BNs through the variational message passing algorithm.[42]

Algorithm 1 describes VB-SEM, which performs as follows. It receives a data set $\mathcal{D}$, an initial model $\mathcal{B}_0$, and a set of arc restrictions $\mathcal{R}_A$ that may be provided by the user or by a related algorithm (see Section 3). Arc restrictions limit the structures produced by the algorithm (e.g., we may not be interested in adding/removing certain graph arcs). Line 1 is the main loop of the algorithm. Line 2 describes the expectation step, in which the current model $\mathcal{B}_t$ estimates the values of its latent variables and generates a completed data set $\mathcal{D}_t^*$. Lines 3–7 describe the maximization step. In Line 3, the completed data set is used to learn the structure $\mathcal{G}_{t+1}$ of the new model $\mathcal{B}_{t+1}$. For simplicity, we use a greedy search (GS) method with arc addition, arc removal, and arc reversal operators.[43] Once the structure has been learned, the parameters θ_{t+1} of the new model are estimated using the observed data $\mathcal{D}$ (Line 4). To this purpose, the VB-EM algorithm[44] is employed. Finally, the resultant model $\mathcal{B}_{t+1}$ is compared with the current best model $\mathcal{B}_t$, and the model with greater score with respect to $\mathcal{D}$ is selected.

TABLE 1 List of abbreviations and acronyms used in this document

Abbreviation	Explanation
AIC	Akaike information criterion
BIC	Bayesian information criterion
BN	Bayesian network
CaD	Cardinality decrease
CaI	Cardinality increase
CD	Critical difference
CLG	Conditional linear Gaussian
CLI	Conditional latent variable introduction
CPD	Conditional probability distribution
CVPLL	Cross-validated predictive log-likelihood
ELBO	Evidence lower bound
EM	Expectation-maximization
GLFM	General latent feature model
GLSL	Greedy latent structure learner
GS	Greedy search
HI-VAE	Heterogeneous incomplete variational autoencoder
IL	Incremental learner
KL	Kullback-Leibler
LCM	Latent class model
LE	Latent variable elimination
LI	Latent variable introduction
LL	Log-likelihood
MB	Markov blanket
MCMC	Markov chain Monte Carlo
MDS-UPDRS	Movement Disorder Society unified Parkinson's disease rating scale
MEAN	Mean imputation
MICE	Multiple imputation chained equations
MKDE	Mixed kernel density estimation
MPMM	Multipartition mixture model
MSPN	Mixed sum-product networks
NMSS	Nonmotor symptom scale
p-ELBO	Penalized evidence lower bound
PD	Parkinson's disease
SD	Standard deviation

(Continues)

TABLE 1 (Continued)

Abbreviation	Explanation
SEM	Structural expectation-maximization
uk-DB	Unsupervised k-dependence Bayesian classifier
VB	Variational Bayes
VB-EM	Variational Bayesian expectation-maximization
VB-SEM	Variational Bayesian-structural expectation-maximization
VI	Variational inference

VB-SEM estimates the parameters and score of the new model with respect to the observed data (i.e., the observed score) rather than with the completed data (i.e., the expected score) to improve its model fitting with respect to the observed data. Despite the desirable properties of SEM, Equation (4) offers no guarantee that the model selected at the end of the search is near the optimum of the observed score.[19] For example, if a search method is used inside SEM, only the first structure change (e.g., an arc addition) guarantees an improvement with respect to the observed score. Benjumeda et al.[45] address this problem by estimating the observed score after each structure change at the expense of the score decomposition. Their approach applies to categorical data, where exact inference can be made tractable by bounding the treewidth of the BN.[46] We instead consider the approximation of estimating the expected score when learning the BN structure, and the observed score when learning the parameters.

Algorithm 1: VB-SEM

Input : $\mathcal{D}, \mathcal{B}_0, \mathcal{R}_A$

for $t = 0, 1, \ldots$ **do**
  /* Expectation step */
  $\mathcal{D}_t^* \leftarrow$ Complete latent data using variational inference with $\mathcal{B}_t$
  /* Maximization step */
  $\mathcal{G}_{t+1} \leftarrow \text{GS}(\mathcal{D}_t^*, \mathcal{R}_A)$
  $\boldsymbol{\theta}_{t+1} \leftarrow \text{VB-EM}(\mathcal{D}, \mathcal{G}_{t+1})$
  $\mathcal{B}_{t+1} \leftarrow \{\mathcal{G}_{t+1}, \boldsymbol{\theta}_{t+1}\}$
  if $score(\mathcal{B}_{t+1} : \mathcal{D}) \leq score(\mathcal{B}_t : \mathcal{D})$ **then**
    break /* Stop the loop */

Output: The resulting model $\mathcal{B}_t$

The scoring function that is traditionally used within the VB framework is the evidence lower bound (ELBO). The ELBO function evaluates how closely the variational distribution $Q(\mathbf{H}, \boldsymbol{\theta})$ approximates the posterior distribution of latent variables and parameters $P(\mathbf{H}, \boldsymbol{\theta}|\mathcal{D}, \mathcal{G})$ using the inverse Kullback–Leibler (KL) divergence:

$$\text{ELBO}(\mathcal{B} : \mathcal{D}) = \log P(\mathcal{D}|\mathcal{G}) - \text{KL}(Q(\mathbf{H}, \boldsymbol{\theta})||P(\mathbf{H}, \boldsymbol{\theta}|\mathcal{D}, \mathcal{G})), \tag{5}$$

where $\log P(\mathcal{D}|\mathcal{G})$ is the logarithm of the evidence (i.e., the marginal-likelihood). However, when applied to models with categorical latent variables, an adjustment must be made to the ELBO to account for the model parameters' lack of identifiability. A common solution involves introducing a small penalty in the ELBO that considers the cardinality k_i of each latent variable H_i in the model.[47] This results in a penalized version of the ELBO (p-ELBO), which we use as our scoring function:

$$\text{p-ELBO}(\mathcal{B} : \mathcal{D}) = \text{ELBO}(\mathcal{B} : \mathcal{D}) - \sum_{i=1}^{s} \log k_i!. \tag{6}$$

Finally, a key aspect for the performance of VB-SEM is prior specification. When expert information is available, prior parameters can be selected to best reflect the expert knowledge. When this information is unavailable, we propose to use the following strategy: (i) observed variables assume empirical Bayes[48] priors (with maximum-likelihood estimates as the values of prior parameters), and (ii) latent variables assume a symmetric Dirichlet prior with a total concentration of 1.00.

2.1 | Complexity analysis

The computational complexity of VB-SEM is directly associated to the computational complexities of the GS (Line 3) and VB-EM (Line 4) algorithms. GS is an iterative algorithm whose computational complexity is quadratic in the number of variables. VB-EM is also an iterative algorithm whose computational complexity is NP-hard due to the use of VI in its expectation step.[49] The latent data completion process (Line 2) is equivalent to a single expectation step of the VB-EM algorithm. Therefore, it is also NP-hard. As a result, the computational complexity of VB-SEM is NP-hard. Nevertheless, we can limit the computational complexity of VB-SEM by establishing a maximum number of iterations for it, for GS, and for VB-EM.

3 | GREEDY LATENT STRUCTURE LEARNER

In this section, we propose a greedy search method for learning CLG BNs with categorical latent variables. Our method iteratively explores this space of models using five latent operators and the VB-SEM algorithm. Latent operators are tasked with introducing latent variables, removing latent variables, and changing the cardinality of latent variables. Each application of the latent operators produces a candidate model whose structure is subsequently refined using a local run of VB-SEM, which introduces, removes, or reverses BN arcs. Both aspects of the structure search (latent variables and BN arcs) are separated to reduce the computational cost of the algorithm.

We start with a description of the latent operators in Section 3.1. Then, in Section 3.2, we present a local version of the VB-SEM algorithm. Finally, in Section 3.3, we discuss the search procedure.

3.1 | Latent operators

- *Latent variable introduction* (LI) generates a new model by introducing a new categorical latent variable H as the parent of two variables (observed or latent) that currently have no

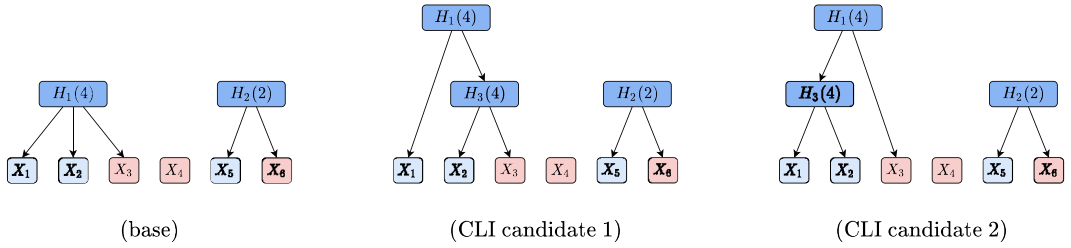

FIGURE 1 Example application of the CLI operator. The base model contains two latent variables (i.e., H_1 and H_2). However only H_1 has more than two children. As a result, two candidate models are generated by introducing a new latent variable H_3 as the child of H_1. Categorical variables are colored blue while continuous variables are colored red. The number in parentheses accompanying each latent variable represents its cardinality. CLI, conditional latent variable introduction [Color figure can be viewed at wileyonlinelibrary.com]

parents. The cardinality of H is set to two. We only consider pairs of variables to reduce the computational complexity of this operator. This restriction will be compensated by the local VB-SEM algorithm.

- *Conditional latent variable introduction* (CLI) produces a new model by introducing a new latent variable H' as the parent of two variables (observed or latent) that currently have a latent variable H as their parent. H' becomes the new child of H. The cardinality of H' is set equal to the cardinality of H. This operator is not applicable if H only has two children. Figure 1 provides an example application of this operator.
- *Latent variable elimination* (LE) generates a new model by removing a latent variable and its associated arcs.
- *Cardinality increase* (CaI) creates a new model by increasing the cardinality of a categorical latent variable H by one. This operator is not applicable if the variable has a cardinality equal to $k_{\max}$, which is established by the user.
- *Cardinality decrease* (CaD) produces a new model by decreasing the cardinality of a categorical latent variable H by one. This operator is not applicable if H already has a cardinality of two.

3.2 | Local VB-SEM

Repeatedly evaluating candidate models produced by the latent operators can become prohibitive when each evaluation involves a full execution of VB-SEM. Therefore, we propose replacing VB-SEM with a local version of this method, which only considers the section of the model that is affected by the latent operator while the rest of the model is kept unchanged. We refer to this method as local VB-SEM.

In the local VB-SEM algorithm, we allow adding, removing, or reversing arcs that contain any of the variables considered by the latent operator. Once the structure has been learned, the local VB-EM algorithm[50] estimates the parameters of those variables belonging to the Markov blankets (MBs) of: (i) variables initially selected by the latent operator, and (ii) variables affected by a structure change (e.g., a new incoming arc). For any variable in a BN, its MB consists of the set of all its parents, children, and spouses (parents of children) in the network. Therefore, the pseudocode of local VB-SEM is identical to the pseudocode of VB-SEM (Algorithm 1), except that the structure learning process is always restrained by a

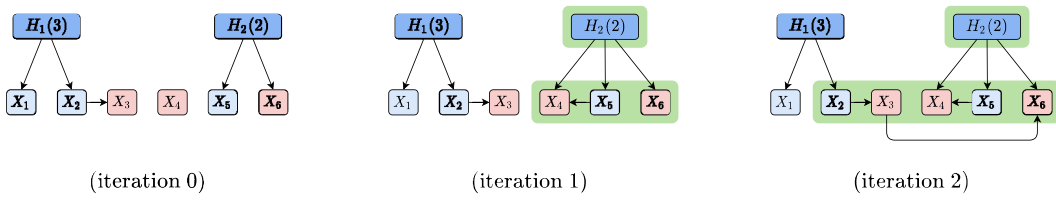

FIGURE 2 Example execution of the local VB-SEM algorithm with two iterations. The initial model is a candidate of the latent introduction operator, which has introduced H_2. It introduces two arcs in its first iteration ($H_2 \rightarrow X_4$ and $X_5 \rightarrow X_4$) and one arc in its second iteration ($X_3 \rightarrow X_6$). Variables highlighted in green have their parameters estimated by the local VB-EM algorithm. Variable colors and parentheses have the same meaning as in Figure 1. VB-SEM, variational Bayesian-structural expectation–maximization [Color figure can be viewed at wileyonlinelibrary.com]

specific set of arc restrictions, and the parameter learning process is done with the local VB-EM algorithm.

One iteration of local VB-SEM is computationally much cheaper than one iteration of VB-SEM because it considers fewer structure changes and it updates fewer model parameters. This also implies that a run of local VB-SEM usually requires fewer steps to converge than a run of VB-SEM. The computational complexity of local VB-SEM is upper-bounded by the computational complexity of VB-SEM.

Figure 2 provides an example execution of the local VB-SEM algorithm. The initial model is a candidate of the LI operator, which has introduced a new latent variable H_2 as the parent of the observed variables X_5 and X_6. In its first iteration, local VB-SEM introduces two new arcs: $H_2 \rightarrow X_4$ and $X_5 \rightarrow X_4$. After the structure learning process, the local VB-EM algorithm estimates the parameters of H_2, X_4, X_5, and X_6 (highlighted in green in the figure), all of which belong to the MBs of the variables initially selected by the latent operator. In its second and last iteration, local VB-SEM introduces a new arc from X_3 to X_6 and estimates the parameters of H_2, X_2, X_3, X_4, X_5, and X_6. While X_2 does not belong to the MB of the variables initially considered by the latent operator, it does belong to the MB of a variable affected by a structure change (i.e., X_3). This is not the case of H_1 and X_1, whose parameters are kept unchanged.

3.3 | Search procedure

The search starts with an initial model $\mathcal{B}_0$ established by the user. Given this model, let $\mathbf{W}$ denote the set of variables that are considered by the LI operator (i.e., the set of variables that currently have no parents) and let $\mathbf{H}_C$ denote the set of latent variables that are considered by the CLI operator (i.e., the set of latent variables that currently have three or more children). Once the initial model has been established, the search method traverses the space of models by iteratively applying the latent operators. Each application of a latent operator produces a set of candidate models that are evaluated using the local VB-SEM algorithm with p-ELBO as the scoring function (see Equation 6). Once all candidate models have been evaluated, the highest scoring model $\mathcal{B}'$ is selected. If its score is higher than that of $\mathcal{B}$, the new model takes its place and the process is repeated. Once the search stops, the resulting model is refined using a full run of VB-SEM. This search procedure is formally defined in Algorithm 2.

Algorithm 2: Greedy latent structure learner (GLSL)

Input : $\mathcal{D}$, $\mathcal{B}_0$, k_{max}, $\mathcal{R}_A$

$\mathcal{B} \leftarrow \mathcal{B}_0$
Let $\mathbf{W}$ be the set of variables that currently have no parent in $\mathcal{B}$
Let $\mathbf{H}_C$ be the set of latent variables with 3 or more children in $\mathcal{B}$
while *True* **do**
  $\mathbf{B}_{LI} \leftarrow \text{LI}(\mathcal{B}, \mathbf{W}, \mathcal{D}, \mathcal{R}_A)$
  $\mathbf{B}_{CLI} \leftarrow \text{CLI}(\mathcal{B}, \mathbf{H}_C, \mathcal{D}, \mathcal{R}_A)$
  $\mathbf{B}_{LE} \leftarrow \text{LE}(\mathcal{B}, \mathcal{D}, \mathcal{R}_A)$
  $\mathbf{B}_{CaI} \leftarrow \text{CaI}(\mathcal{B}, \mathcal{D}, k_{max}, \mathcal{R}_A)$
  $\mathbf{B}_{CaD} \leftarrow \text{CaD}(\mathcal{B}, \mathcal{D}, \mathcal{R}_A)$
  $\mathcal{B}' \leftarrow$ highest scoring model in $\{\mathbf{B}_{LI}, \mathbf{B}_{CLI}, \mathbf{B}_{LE}, \mathbf{B}_{CaI}, \mathbf{B}_{CaD}\}$
  if $score(\mathcal{B}' : \mathcal{D}) > score(\mathcal{B} : \mathcal{D})$ **then**
    $\mathcal{B} \leftarrow \mathcal{B}'$
    Update $\mathbf{W}$ and $\mathbf{H}_C$ with respect to $\mathcal{B}$
  else
    break `/* Stop the loop */`
`/* Model refinement */`
$\mathcal{B}' \rightarrow$ VB-SEM($\mathcal{D}$, $\mathcal{B}$, $\mathcal{R}_A$)

Output: The resulting model $\mathcal{B}'$

Figure 3 provides an example execution of the GLSL algorithm. It starts with an empty graph with three categorical variables $\{X_1, X_2, X_5\}$ and three continuous variables $\{X_3, X_4, X_6\}$. In its first iteration, the LI operator is selected and a new latent variable H_1 is introduced as the parent of X_3 and X_4. In addition, the local run of VB-SEM results in other three arcs: $H_1 \rightarrow X_1$, $H_1 \rightarrow X_2$, and $X_5 \rightarrow X_4$. In its second iteration, the CaI operator is selected, and the cardinality of H_1 is increased. No structure changes are introduced by the local VB-SEM. In its third iteration, the CLI operator is selected. A new latent variable H_2 is introduced as the child of H_1, and as the parent of X_3 and X_4. In the fourth and last iteration, the CaD operator is selected, and the cardinality of H_2 is decreased. In addition, the local run of VB-SEM introduces a new arc from X_1 to X_3. Finally, the refinement run of VB-SEM introduces a new arc from X_5 to X_6 and the model is returned.

3.3.1 | Complexity analysis

The following auxiliary variables are considered for the complexity analysis of GLSL:

- $\mathbf{W}\rightarrow$ current set of variables with no parents.
- $\mathbf{H}\rightarrow$ current set of latent variables.
- $\mathbf{H}_C\rightarrow$ current set of latent variables with three or more children.
- $\mathbf{Ch}_{Ci}\rightarrow$ current set of children variables of the latent variable $H_i \in \mathbf{H}_C$.

At each iteration of the GLSL algorithm, the CaI, CaD, and LE operators evaluate a total of $O(|\mathbf{H}|)$ candidate models each. In turn, the LI operator evaluates a total of $O((|\mathbf{W}|^2 - |\mathbf{W}|)/2)$ candidate models. Estimating the number of candidate models that are evaluated by the CLI

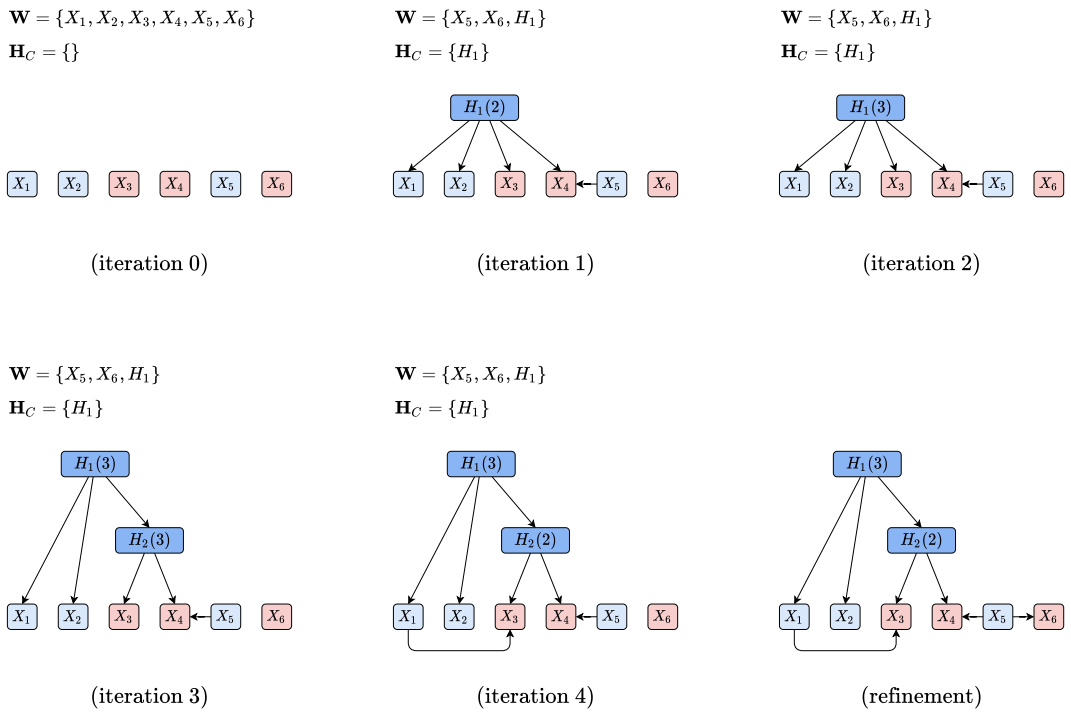

FIGURE 3 Example execution of the GLSL algorithm with four iterations. It introduces a latent variable H_1 in the first iteration, it increases the cardinality of H_1 in the second iteration, it introduces a new conditional latent variable H_2 in the third iteration, and it decreases the cardinality of H_2 in the fourth iteration. Each iteration is followed by a local run of the VB-SEM algorithm, which is tasked with introducing, removing, or reversing BN arcs. Once the iteration process of GLSL finishes, a full run of VB-SEM refines the structure, introducing a new arc from X_5 to X_6. Variable colors and parentheses have the same meaning as in Figure 1. GLSL, greedy latent structure learner; VB-SEM, variational Bayesian-structural expectation–maximization [Color figure can be viewed at wileyonlinelibrary.com]

operator is more complex than for LI. This is because CLI depends on the current number of latent variables and their respective number of children variables. The CLI operator evaluates a total of $O(\sum_i(|\mathbf{Ch}_{Ci}|^2 - |\mathbf{Ch}_{Ci}|)/2)$ candidate models. Therefore, the total number of candidate models evaluated at each iteration of GLSL is $O(3|\mathbf{H}| + (|\mathbf{W}|^2 - |\mathbf{W}|)/2 + \sum_i(|\mathbf{Ch}_{Ci}|^2 - |\mathbf{Ch}_{Ci}|)/2)$. Note that each evaluation requires running the local VB-SEM algorithm.

4 | EXPERIMENTS

In this section, we first evaluate the performance of GLSL in a clustering task with Parkinson's disease data. Then, we conduct a comparative study with several mixed data sets, where we explore the performance of our method in density estimation tasks. Finally, we apply GLSL to the task of imputing missing data. Experiments were run on a computer with an Intel Core i7-6700K CPU at 4.00 GHz with 64 GB RAM, running Windows 10 Enterprise. GLSL was implemented in Java 8 using the AMIDST toolbox.[51] For reproducibility, all code, data, and results can be found in the following public repository: https://github.com/ferjorosa/mpc-mixed.

4.1 | Multipartition clustering of Parkinson's disease data

In the first experiment, we evaluated the ability of GLSL for multipartition clustering. To this end, we considered data from an international study of 402 Parkinson's disease (PD) patients.[52] For all patients in the study, basic clinical information (i.e., sex, age, age of onset, and disease duration) was recorded along with motor and nonmotor symptoms. Eleven motor symptoms were considered, all of them corresponded to items from the Movement Disorder Society unified Parkinson's disease rating scale (MDS-UPDRS).[53] The severity of each motor symptom was rated with a categorical value of five possible outcomes: normal, slight, mild, moderate, and severe. Eleven nonmotor symptoms were also considered, all of them corresponded to domains from the Nonmotor Symptom Scale (NMSS).[54] The severity of each nonmotor symptom was rated with a continuous value between 0.00 and 12.00. Tables 2–4 provide a summary of the data. In addition, 62% of the patients in the study were male.

The study was approved by the institutional review boards or ethics committees of the participating centers, and the study was conducted according to good clinical practice. All patients gave their written consent to participate in the study. Institutional review board or ethics committee that approved the study: (1) National Research Ethics Service Committee East Midlands—Northampton, England; (2) Institutional Review Board at the Perelman School of Medicine at the University of Pennsylvania, United States.

TABLE 2 Basic clinical information of Parkinson's disease patients

	Mean	*SD*	Median	Min	Max
Age	67.42	9.96	68	35	93
Age of onset	59.23	10.67	59	26	93
Disease duration	8.19	5.93	7	0	35

Abbreviations: Max, maximum recorded value; Min, minimum recorded value; *SD*, standard deviation.

TABLE 3 Motor information of Parkinson's disease patients

	Normal (%)	Slight (%)	Mild (%)	Moderate (%)	Severe (%)
Speech	34	40	22	4	0
Rigidity	11	44	32	11	2
Gait	21	51	18	9	1
Freezing	81	10	6	3	0
Postural instability	46	25	15	12	2
Postural tremor	43	39	15	3	0
Kinetic tremor	52	34	13	1	0
Resting tremor	45	27	23	5	0
Bradykinesia	43	30	19	7	1
Dyskinesias	75	17	4	2	2
Motor fluctuations	46	21	13	8	12

TABLE 4 Nonmotor information of Parkinson's disease patients

	Mean	*SD*	Median	Min	Max
Cardiovascular	0.62	1.16	0.00	0.00	9.00
Fatigue	1.96	2.08	1.50	0.00	12.00
Mood	1.10	1.82	0.34	0.00	10.00
Hallucinations	0.35	0.92	0.00	0.00	7.00
Attention	1.43	1.96	0.67	0.00	10.00
Gastrointestinal	1.26	1.74	0.33	0.00	10.00
Urinary	2.15	2.61	1.33	0.00	12.00
Sexual	1.64	2.99	0.00	0.00	12.00
Hyposmia	4.15	4.69	3.00	0.00	12.00
Weight change	0.69	2.26	0.00	0.00	12.00
Sweating	0.86	2.40	0.00	0.00	12.00

Abbreviations: Max, maximum recorded value; Min, minimum recorded value; *SD*, standard deviation.

The main idea behind this experiment was using motor and nonmotor information to find homogeneous clusters of PD patients across different subsets of related symptoms. Then, analyze the resulting clusters using basic clinical information. Clusters of PD patients are usually considered a subtype of the disease.[55] Identification of PD subtypes may help understand underlying disease mechanisms, since homogeneous clusters of patients may be more likely to share pathological and genetic features. In addition, identification of PD subtypes may ultimately lead to more precise treatment strategies.[56]

To explore the relationship between basic clinical information and the identified clusters of patients, hypothesis tests were performed. Each pair of clusters in a partition were compared. For continuous variables such as age, age of onset, and PD duration, a Mann–Whitney U test[57] was used. When three or more clusters were present in a partition, a Kruskal–Wallis test[58] was performed instead, followed by a post-hoc analysis using Tukey's range test.[59] For categorical variables such as the sex of the patient, χ^2 tests were performed. Statistical significance was defined as $p < 0.05$. To perform these tests, probabilistic clusters were transformed into non-probabilistic clusters by assigning each patient to its most probable cluster in each partition.

4.1.1 | Methods

Due to the absence of prior information, we considered empirical Bayes priors for GLSL. We initialized GLSL with a BN structure where all the observed variables were independent (i.e., an empty graph). In addition, we did not contemplate arc restrictions besides those intrinsic to CLG BNs. We compared GLSL with several model-based clustering methods from the literature that allow mixed data. Given the low number of multipartition clustering methods that allow mixed data, we considered four single-partition clustering methods (i.e., traditional model-based clustering methods that learn a mixture model with a single latent variable) and one multipartition clustering method:

- Latent class model (LCM),[60] which learns a mixture model where all the observed variables are conditionally independent given a categorical latent variable. The cardinality of the latent variable is iteratively estimated by maximizing the BIC score. The implementation is provided in our public repository.
- Unsupervised k-dependence Bayesian classifier (uk-DB),[61] which learns a BN model with a single categorical latent variable that is the parent of all the observed variables. Observed variables may also be the children of other $k - 1$ observed variables. The cardinality of the latent variable is iteratively estimated using a scoring function and the arcs between observed variables are usually estimated using SEM. Given the presence of mixed data, we used the p-ELBO and the VB-SEM algorithm for the structure learning process. We also considered $k = 3$. The implementation is provided in our public repository.
- ClustMD,[62] which learns a mixture of latent variable models, where categorical variables are internally transformed to the continuous domain. We used the implementation provided by the public R package (https://cran.r-project.org/web/packages/clustMD/).
- MixCluster,[63] which learns a Gaussian copula mixture model that defines intra-component dependencies similar to a Gaussian mixture model. We used the R implementation available at https://rdrr.io/rforge/MixCluster/.
- Multipartition mixture model (MPMM),[15] which learns a mixture model with several categorical latent variables. Each latent variable is the parent of a unique subset of observed variables. Each observed variable is conditionally independent of the rest of variables given its latent parent. The implementation is provided in our public repository.

A comparison of the computational complexity of these algorithms and others from the current multipartition clustering literature is provided as online supporting information (see Table S2).

4.1.2 | Results

To evaluate the quality of the considered methods, we estimated the LL and BIC scores of their respective models. As indicated by Table 5, the scores of single-partition clustering models (i.e., LCM, uk-DB, and ClustMD) were similar between them but far from those of multipartition clustering models (i.e., MPMM and GLSL). All of the considered methods were able to finish execution in reasonable time except MixCluster, which could not finish even after 96 h of execution. In terms of score, GLSL returned the best model, showing the highest LL and BIC scores.

In terms of clustering quality, we observed remarkable differences between the results of single-partition methods and multipartition methods. All single-partition methods were only able to find three clusters of patients, which could be easily associated with the general severity of all the disease symptoms. Alternatively, multipartition methods were able to identify subsets of related symptoms where specific clusters of PD patients could be found. In addition to fitting data more closely, multipartition methods were able to identify a higher number of clusters. MPMM identified 13 clusters across four latent variables. We refer to each latent variable as a partition. The first partition divided patients into two clusters by considering motor aspects such as posture, dyskinesias, and freezing. The second partition divided patients into three clusters according to the intensity of their tremors. The third partition divided patients into four clusters according to gait, motor fluctuations, and hyposmia. The fourth partition divided

TABLE 5 Performance of clustering methods on the Parkinson's data

	LL[a]	BIC[a]	Time (s)	#Partitions	#Clusters	#Parameters
LCM	−12,496.18	−13,197.77	10.07	1	3	234
u*k*-DB	−11,452.64	−15,257.39	1458.72	1	3	1269
ClustMD	−12,613.72	−13,009.48	**5.35**	1	3	132
MPMM	−10,433.75	−11,186.30	4081.22	4	13	251
GLSL	**−8877.02**	**−9656.55**	1118.59	9	19	260

Note: The winner in each column is indicated in bold.

Abbreviations:#Clusters, total number of clusters across all partitions; #Parameters, number of model parameters; #Partitions, number of partitions; BIC, Bayesian information criterion; LL, log-likelihood; s, seconds.

[a]The higher the score, the better.

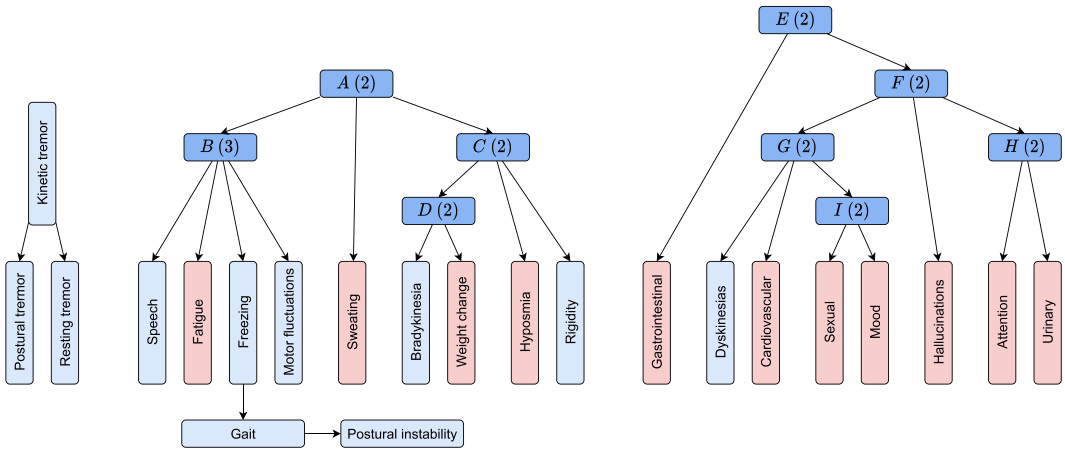

FIGURE 4 Bayesian network structure of the multipartition clustering model returned by the Greedy latent structure learner algorithm. Colors and parentheses have the same meaning as in Figure 1 [Color figure can be viewed at wileyonlinelibrary.com]

patients into four clusters according to all the nonmotor symptoms. Alternatively, GLSL found a total of 19 clusters across nine partitions. They are described in the following section. The BN structure that resulted from applying GLSL is portrayed in Figure 4. For comparison, the BN structures that resulted from applying LCM, u*k*-DB and MPMM are provided as online supporting information (see Figures S1–S3). ClustMD is not included because its result does not have a direct BN representation.

4.1.3 | GLSL partitions

GLSL identified nine (alphabetically named) latent variables. Each latent variable provided a different way to partition PD patients according to a specific set of motor and/or nonmotor variables. The sex, age, age of onset, and disease duration of each cluster of patients in a partition is provided in Table 6. Statistically significant differences between clusters are

TABLE 6 Basic clinical information of each cluster of patients

	Cluster	Sex (% male)	Age	Age of onset	Duration
Partition *A*	*A*1	64	67.94 (9.79)[a]	59.96 (10.44)[a]	7.98 (5.90)[a]
	*A*2	52	64.91 (10.44)[a]	55.68 (11.14)[a]	9.23 (6.02)[a]
Partition *B*	*B*1	65	67.05 (10.30)	60.37 (11.01)[b2]	6.68 (5.07)[b1]
	*B*2	62	68.28 (9.73)	58.91 (10.61)	9.37 (6.33)[b1]
	*B*3	50	64.09 (8.93)	55.00 (7.96)[b2]	9.09 (6.02)
Partition *C*	*C*1	64	67.97 (10.22)	60.01 (10.50)[c]	7.96 (6.16)
	*C*2	61	66.97 (9.74)	58.00 (10.80)[c]	8.39 (5.74)
Partition *D*	*D*1	63	67.21 (10.09)	59.18 (10.60)	8.03 (5.86)
	*D*2	54	68.74 (9.07)	59.51 (11.22)	9.23 (6.28)
Partition *E*	*E*1	60	66.05 (10.44)[e]	58.92 (11.02)	7.13 (5.65)[e]
	*E*2	66	69.22 (9.01)[e]	59.63 (10.22)	9.60 (6.01)[e]
Partition *F*	*F*1	61	67.10 (10.27)	59.59 (10.90)	7.51 (5.67)[f]
	*F*2	65	68.52 (8.78)	58.00 (9.84)	10.52 (6.22)[f]
Partition *G*	*G*1	61	67.00 (10.07)	59.30 (11.02)	7.70 (5.59)[g]
	*G*2	64	68.13 (9.76)	59.10 (10.12)	9.03 (6.39)[g]
Partition *H*	*H*1	59	66.52 (9.85)	59.06 (10.78)	7.46 (5.86)[h]
	*H*2	65	68.29 (10.01)	59.39 (10.60)	8.90 (5.93)[h]
Partition *I*	*I*1	57[i]	67.05 (10.94)	59.67 (11.45)	7.38 (5.51)[i]
	*I*2	68[i]	67.81 (8.83)	58.77 (9.82)	9.05 (6.24)[i]

Note: Numbers between parentheses correspond to *SD*s.

[a]Statistically significant differences ($p < 0.05$) between *A*1 and *A*2.

[b1]Statistically significant differences ($p < 0.05$) between *B*1 and *B*2.

[b2]Statistically significant differences ($p < 0.05$) between *B*1 and *B*3.

[c]Statistically significant differences ($p < 0.05$) between *C*1 and *C*2.

[d]Statistically significant differences ($p < 0.05$) between *D*1 and *D*2.

[e]Statistically significant differences ($p < 0.05$) between *E*1 and *E*2.

[f]Statistically significant differences ($p < 0.05$) between *F*1 and *F*2.

[g]Statistically significant differences ($p < 0.05$) between *G*1 and *G*2.

[h]Statistically significant differences ($p < 0.05$) between *H*1 and *H*2.

[i]Statistically significant differences ($p < 0.05$) between *I*1 and *I*2.

included. Results (i.e., *p* values) of the statistical tests that were performed are provided as online supporting information (see Table S1).

In partition *A*, patients were divided into two clusters according to the severity of their sweating problems. A total of 82% of patients (cluster *A*1) did not show any sweating problems (mean of 0.00), while the remaining 18% of the patients (cluster *A*2) showed mild sweating problems (mean of 4.99). We observed that patients in *A*2 were significantly younger, had a significant younger age of onset, and presented a significantly longer disease duration than patients in *A*1.

Partition *B* distinguished three clusters of patients that differed according to their freezing, gait, postural instability, fatigue, motor fluctuations, and speech problems. Gait and postural

instability were indirectly associated with the rest of the partition symptoms via the freezing symptom. We observed a direct correspondence between freezing, gait and postural instability. The first cluster ($B1$) consisted of 38% of the patients, and showed almost no presence of the considered symptoms. The second cluster ($B2$) consisted of 52% of the patients, which presented a higher degree of fatigue, speech problems and motor fluctuations than patients from cluster $B1$, but not much freezing, gait, or postural instability. The third cluster ($B3$) consisted of 10% of the patients, which showed all the considered symptoms with greater intensity (including freezing, gait, and postural instability) compared to $B1$ and $B2$. We observed that patients in $B3$ had a significantly younger age of onset than patients in $B1$. We also observed that patients in $B2$ had a significantly longer disease duration than patients in $B1$. However, we did not observe differences between $B2$ and $B3$ that were statistically significant.

Rigidity and hyposmia were associated in partition C. Two clusters of patients were identified, where the $C1$ consisted of 45% of the patients that did not suffer hyposmia (mean of 0.00) and presented a slight-mild rigidity. The second cluster ($C2$) was formed by the remaining 55% of the patients, which suffered moderate hyposmia (mean of 7.55) and a higher degree of rigidity. We observed that patients in $C2$ had a significantly younger age of onset than those in $C1$.

Bradykinesia was associated with weight changes in partition D. Two clusters of patients could be distinguished. The first cluster ($D1$) was formed by 85% of the patients, which presented no sudden changes of weight (mean of 0.00) and slight-mild bradykinesia. The second cluster ($D2$) was composed of the remaining 15% of patients, which showed considerable weight changes (mean of 4.85) and mild–moderate bradykinesia. We did not observe statistically significant differences between $D1$ and $D2$ in the basic clinical variables of Table 6.

In partition E, patients were classified into two clusters according to their gastrointestinal issues. Patients in $E1$ (52%) presented low-intensity gastrointestinal problems (mean of 0.12) and patients in $E2$ (48%) presented gastrointestinal problems with much greater intensity (mean of 2.49). We observed that patients with a higher degree of gastrointestinal problems ($E2$) were significantly older and had a higher disease duration.

Partition F divided patients into two clusters according to the intensity of their hallucinations. A total of 76% of the patients formed the first cluster ($F1$), which did not suffer any remarkable hallucinations (mean of 0.00). The second cluster ($F2$) was formed by the remaining 24% of the patients and suffered slight hallucinations (mean of 1.46). We observed a relationship between the presence of hallucinations and the duration of the disease, where patients in $F2$ had a significantly longer PD duration than patients in $F1$.

Dyskinesias and cardiovascular problems were related in partition G, which established two clusters of patients. The first cluster ($G1$) consisted of 61% of the patients, presenting slight–mild dyskinesias and no cardiovascular problems (mean of 0.00). The second cluster ($G2$) was formed by the remaining 39% of the patients. Patients in $G2$ presented a higher degree of dyskinesias and were also affected by cardiovascular problems (mean of 1.60). We observed that patients in $G2$ had a significantly longer duration of PD than patients in $G1$.

In partition H, patients were divided into two clusters according to the intensity of their attention and urinary problems. 47% of the patients formed $H1$, which was characterized by attention (mean of 0.25) and urinary (mean of 0.66) problems with a very low intensity. The remaining 53% of the patients formed $H2$, which was characterized by more intense attention (mean of 2.52) and urinary (mean of 3.52) problems. We observed that patients in $H2$ had a significantly longer duration of the disease than patients in $H1$.

Sexual problems were associated with mood problems in partition I. Two clusters of patients were identified. The first cluster ($I1$) was constituted by 51% of the patients, who

showed no sexual problems (mean of 0.00) and almost no mood problems (mean of 0.33). The second cluster ($I2$) was formed by the remaining 49% of the patients and showed considerable sexual (mean of 3.35) and mood (mean of 1.92) problems. We observed that patients in $I2$ were significantly male and had a significantly longer duration of PD than patients in $I1$.

Tremor variables were associated with each other without a latent variable. There was a direct correspondence between the states of tremor variables (i.e., patients without kinetic tremors had an increased probability of not suffering postural and resting tremors, patients with slight kinetic tremors had an increased probability of suffering postural and resting tremors, etc.). This suggested that patients could be grouped according to tremor into five groups, one for each state of the tremor variables.

Finally, it is important to note that several associations between partitions were identified. Belonging to a specific cluster in a partition influenced the cluster probabilities in other partitions. By using probabilistic inference, we were able to study the effect of these associations on their respective symptoms. Some interesting patterns that we observed included: (i) patients with a higher degree of dyskinesias and cardiovascular problems (cluster $G2$) had a 0.64 probability of suffering sexual and mood problems (cluster $I2$); (ii) patients that had experienced mild–moderate bradykinesia and considerable weight changes (cluster $D2$) had a 0.79 probability of showing moderate hyposmia and a higher degree of rigidity (cluster $C2$); and (iii) patients that did not suffer attention and urinary problems (cluster $H1$) had an 0.85 probability of not suffering hallucinations (cluster $F1$). In this analysis, probabilistic inferences were carried out using Monte Carlo sampling in the tool GeNIe (https://www.bayesfusion.com/genie/). A total of 29 probabilistic queries were performed to analyze the associations between the nine partitions. The results of these queries are provided as online supporting information (see Table S1). Furthermore, the multipartition clustering model in GeNIe format is also available in our code repository.

4.2 | Comparative study on density estimation

In the second experiment, we evaluated the performance of GLSL in terms of data fitting and computational cost. To conduct the study, we used 36 real-world data sets of varied dimensionalities and sample sizes. The majority of these data sets were retrieved from the UCI repository.[64] We also added several data sets that were used by methods from the literature (i.e., hiv-test,[5] alarm,[6] and NBA[10]). We summarize the main characteristics of these data sets in Table 7.

As a performance measure, we used the ten-fold cross-validated predictive log-likelihood (CVPLL). That is, we divided each data set into 10 equal-sized folds, trained a model on nine of them, and computed the predictive log-likelihood on the remaining fold. We did this for each fold and averaged the results.

We analyzed the significance of the differences found in the study using the Friedman test[65] with $\alpha = 0.05$ and Nemenyi's post-hoc test.[66] According to the Nemenyi's test, the difference in performance of two methods is statistically significant if their corresponding mean ranks differ by at least the critical difference (CD):

$$CD = q_\alpha \sqrt{\frac{k(k+1)}{6n}}, \tag{7}$$

TABLE 7 Basic properties of the considered data sets

Data set	*m*	#Categorical	#Continuous	*N*
HIV-test	4	4	0	428
Hayes-roth	5	5	0	160
Balance-scale	5	5	0	625
Somerville	7	7	0	143
Car-evaluation	7	7	0	1728
Nursery	9	9	0	12960
Breast-cancer	10	10	0	277
Web-phishing	10	10	0	1353
Zoo	17	17	0	101
Vote	17	17	0	232
Spect-heart	23	23	0	267
Alarm	36	36	0	1000
Real-state	5	0	5	414
Buddymove	6	0	6	249
Qsar-fish	7	0	7	908
Qsar-aqua	9	0	9	545
ILPD	9	0	9	579
Alcohol	10	0	10	125
Travel-reviews	10	0	10	980
Wine-quality	12	0	12	4898
Wine	13	0	13	178
Leaf	14	0	14	340
NBA	18	0	18	441
WDBC	30	0	30	569
Haberman	4	1	3	306
Iris	5	1	4	150
Blood-transfusion	5	1	4	748
User-knowledge	6	1	5	258
Vertebral	7	1	6	310
Ecoli	7	1	6	336
Planning-relax	12	1	12	4898
Thoracic-surgery	14	11	3	470
Vehicle	19	1	18	846
Thyroid	21	16	5	3103

(Continues)

TABLE 7 (Continued)

Data set	m	#Categorical	#Continuous	N
Parkinsons	23	1	22	195
Ionosphere	34	1	33	351

Note: Data sets are grouped according to the type of their variables and sorted in ascending order according to their dimensionality.

Abbreviations: #Categorical, number of categorical variables; #Continuous, number of continuous variables; m, total number of variables; N, sample size.

where k is the number of algorithms, n is the number of data sets, and q_α corresponds to the Studentized range statistic divided by $\sqrt{2}$.

4.2.1 | Methods

Due to the absence of prior information, we considered empirical Bayes priors for GLSL. We initialized GLSL with a BN structure where all the observed variables were independent (i.e., an empty graph). In addition, we did not contemplate arc restrictions besides those intrinsic to CLG BNs. We compared GLSL with several parametric and nonparametric methods from the current literature:

- Mixed kernel density estimation (MKDE),[67] which is a variant of the multivariate kernel density estimator that can deal with mixed data, and uses Silverman's rule of thumb for bandwidth estimation. We used the implementation provided by the Python library Statsmodels (https://www.statsmodels.org/).
- Mixed sum-product networks (MSPN),[68] which combine sum-product networks[69] with nonparametric estimation to learn hierarchically structured latent variable models that do not require the specification of variables' parametric forms. We used the implementation provided by the Python library SPFlow (https://github.com/SPFlow/SPFlow).
- General latent feature model (GLFM),[70] which handles both categorical and continuous variables using a Bayesian nonparametric model with MCMC inference. We ran 1000 iterations of the sampler using the Python implementation available at https://github.com/ivaleraM/GLFM.
- Incremental learner (IL),[50] which hill-climbs the space of latent forests in a two-phase iterative process. In its first phase, the forest structure is incremented with a new arc or latent variable. In its second phase, the cardinalities of latent variables are determined. We used the implementation available at https://github.com/ferjorosa/incremental-latent-forests
- Variational autoencoder for heterogeneous mixed type data (VAEM),[71] which uses a two-stage variational autoencoder that is able to handle both categorical and continuous variables. We adopted the same neural network architecture as in Ma et al.[71] The generator (decoder) is a 20–50–100 fully connected neural network with ReLU activation functions on hidden units. The encoder considers a structure of m-500–200–40 (where m is the number of observed variables) that maps the observed data into distributional parameters of the latent space. We trained the model with minibatches of 1000 samples and 2000 epochs, using the Python implementation available at https://github.com/microsoft/VAEM.

4.2.2 | Results

Tables 8 and 9 display the experimental results of the comparative study. The rank matrix displays the number of times each algorithm ranked first, second, third, and so on. Then, the mean rank of each method over all the data sets is displayed. The ranking of the methods is given by their average performance (CVPLL and time) compared to the others (i.e., the best is ranked first and the worst is ranked sixth). The detailed results are supplied as online supporting information (see Tables S4 and S5).

Figures 5 and 6 present the CD diagrams of the Nemenyi tests for the CVPLL and time results, respectively. In the figures, groups of methods whose differences are not statistically significant are connected with a thick horizontal line. This graphical representation was first proposed by Demšar.[72] Friedman's tests for CVPLL and time returned p-values of 1.61e−15 and 8.60e−33, respectively.

In the case of CVPLL, statistically significant differences were found among the methods. GLSL had the best performance in 17 out of 36 data sets, followed by IL and MKDE, each performing best in 5 out of 36 data sets. Interestingly, VAEM reached first position in 7 out of 36 data sets and showed a great performance in those data sets composed solely of continuous variables. However, its performance was considerably worse when there were categorical variables in the data. In general, GLSL performed significantly better than the rest of methods (excluding IL). These results indicate that approaches based on CLG BNs with categorical

TABLE 8 CVPLL performances in the comparative study on density estimation

	Rank matrix						
	1st	**2nd**	**3rd**	**4th**	**5th**	**6th**	**Mean rank**
MKDE	5	6	6	17	2	0	3.14 (1.20)
MSPN	2	4	6	10	5	9	4.08 (1.52)
GLFM	0	1	1	1	20	13	5.19 (0.86)
IL	5	11	18	2	0	0	2.47 (0.81)
VAEM	7	2	1	5	7	14	4.25 (1.95)
GLSL	17	12	4	1	2	0	**1.86 (1.10)**

Note: Numbers between parentheses correspond to *SD*s; the mean rank winner is indicated in bold.

TABLE 9 Execution time performances in the comparative on density estimation

	Rank matrix						
	1st	**2nd**	**3rd**	**4th**	**5th**	**6th**	**Mean rank**
MKDE	36	0	0	0	0	0	**1.00 (0.00)**
MSPN	0	26	9	1	0	0	2.31 (0.52)
GLFM	0	2	11	23	0	0	3.58 (0.60)
IL	0	8	16	8	4	0	3.22 (0.93)
VAEM	0	0	0	4	25	7	5.08 (0.55)
GLSL	0	0	0	0	7	29	5.81 (0.40)

Note: Numbers between parentheses correspond to *SD*s; the mean rank winner is indicated in bold.

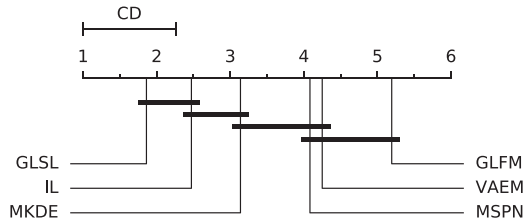

FIGURE 5 Critical difference diagram. Comparison of mean rank CVPLL with Nemenyi's post-hoc procedure in 36 real-world mixed data sets. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; IL, incremental learner; MKDE, mixed kernel density estimation; MSPN, mixed sum-product networks; VAEM, variational autoencoder for heterogeneous mixed

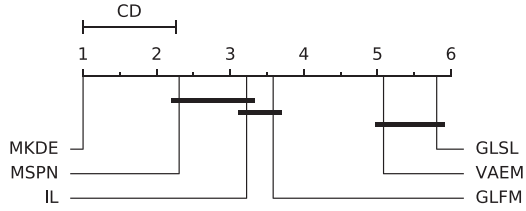

FIGURE 6 Critical difference diagram. Comparison of mean rank execution time with Nemenyi's post-hoc procedure in 36 real-world mixed data sets. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; IL, incremental learner; MKDE, mixed kernel density estimation; MSPN, mixed sum-product networks; VAEM, variational autoencoder for heterogeneous mixed

latent variables (i.e., GLSL and IL) are able to fit mixed data considerably well, surpassing other parametric and nonparametric approaches from the literature. In addition, while not significantly better, the flexibility provided by GLSL resulted in better-fitting models than those produced by IL.

In the case of execution time, MKDE was the undisputed winner, being the fastest method in all of the data sets. MKDE was followed by MSPN, which ended second in 26 out of 36 data sets. GLSL was the slowest method (followed by VAEM), ending last in all of the data sets. However, it should be noted that GLSL explores a large space of models (compared e.g., to IL). In addition, while MKDE, MSPN, GLFM, and VAEM implementations run in parallel, GLSL is currently implemented for single-thread execution.

4.3 | Comparative study on missing data imputation

In the last experiment, we used GLSL to estimate missing data in 36 real-world data sets (see Table 7). For each data set, we generated five different incomplete data sets, removing completely at random a percentage of the data ranging from a 10% deletion to a 50% deletion. As a performance measure, we estimated the average imputation error. Given a data set with N instances and m variables, the average imputation error is computed as follows:

$$AvgErr = \frac{1}{m}\sum_{i} err(i),$$

where we use the following error metrics for each variable X_i, since the computation of the errors depends on the type of variable we are considering:

- Normalized root mean square error for continuous variables, that is,

$$err(i) = \frac{\sqrt{1/N\sum_n (x_i[n] - \hat{x}_i[n])^2}}{max(X_i) - min(X_i)}.$$

- Accuracy error for categorical variables, that is,

$$err(i) = \frac{1}{N}\sum_n \delta(x_i[n] \neq \hat{x}_i[n]).$$

Imputations were done using MAP estimates, where $\hat{x}_i[n]$ refers to the imputed value of X_i in the data instance n, and δ refers to the Kronecker delta. Identically to the density estimation task, we analyzed the significance of the differences found in the study using the Friedman test with $\alpha = 0.05$ and Nemenyi's post-hoc procedure.

4.3.1 | Methods

Due to the absence of prior information, we considered empirical Bayes priors for GLSL. We initialized GLSL with a BN structure where all the observed variables were independent (i.e., an empty graph). In addition, we did not contemplate arc restrictions besides those intrinsic to CLG BNs. We compared GLSL with the following methods:

- *Mean imputation*. Baseline algorithm that imputes the missing data with the mean of each continuous variable and the mode of each categorical variable.
- *MICE*. Multiple imputation by chained equations,[73] which is an iterative method that performs a series of regression models, in which missing data is modeled conditionally on the other variables in the data. We used the implementation provided by the Python library scikit-learn (https://scikit-learn.org/).
- *GLFM*. Gaussian latent feature model.[70]
- *HI-VAE*. Heterogeneous incomplete variational autoencoder,[74] which provides a general framework for variational autoencoders that effectively incorporates incomplete data and handles both categorical and continuous variables. We used the same neural network architecture as in[74] and set the dimensionality of parameters z, y and s to 10, 5 and 10, respectively. We trained the model with minibatches of 1000 samples and 2000 epochs, using the Python implementation available at https://github.com/probabilistic-learning/HI-VAE.

4.3.2 | Results

Tables 10 and 11 illustrate the experimental results of the comparative study. Each method is ranked given their average performance (average imputation error and execution time)

TABLE 10 Average imputation error rank in the comparative study on missing data imputation

Percentage (%)	MEAN	MICE	GLFM	HI-VAE	GLSL
10	4.81 (0.71)	2.08 (1.08)	3.83 (0.65)	2.47 (0.81)	**1.81 (0.92)**
20	4.78 (0.76)	2.25 (1.13)	3.58 (0.87)	2.36 (0.83)	**2.03 (1.18)**
30	4.58 (0.94)	2.64 (1.22)	3.61 (0.93)	2.19 (0.79)	**2.00 (1.31)**
40	4.56 (0.84)	2.86 (1.38)	3.64 (0.76)	2.00 (0.83)	**1.97 (1.21)**
50	4.31 (0.98)	2.67 (1.35)	3.92 (0.91)	**1.97 (0.81)**	2.22 (1.33)

Note: Numbers between parentheses correspond to *SD*s; the mean rank winner is indicated in bold.

TABLE 11 Execution time rank in the comparative study on missing data imputation

Percentage (%)	MEAN	MICE	GLFM	HI-VAE	GLSL
10	**1.00 (0.00)**	2.00 (0.00)	3.36 (0.49)	3.64 (0.49)	5.00 (0.00)
20	**1.00 (0.00)**	2.00 (0.00)	3.33 (0.48)	3.72 (0.57)	4.94 (0.23)
30	**1.00 (0.00)**	2.00 (0.00)	3.33 (0.48)	3.69 (0.52)	4.97 (0.17)
40	**1.00 (0.00)**	2.00 (0.00)	3.36 (0.49)	3.75 (0.60)	4.89 (0.40)
50	**1.00 (0.00)**	2.00 (0.00)	3.31 (0.47)	3.78 (0.59)	4.92 (0.28)

Note: Numbers between parentheses correspond to *SD*s; the mean rank winner is indicated in bold.

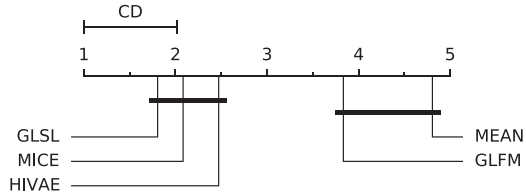

FIGURE 7 Critical difference diagram. Comparison of mean rank average imputation error with Nemenyi's post-hoc procedure when 10% of the data is missing. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; MEAN, mean imputation; MICE, multiple imputation chained equations

compared to the rest. The detailed results are supplied as online supporting information (see Figures S4–S6 and Table S6). In addition, Figures 7–11 display the CD diagrams of the Nemenyi tests for the average imputation error at 10%–50% of missing data. Friedman's tests of these experiments returned *p*-values of 2.26e−19 (10% of missing data), 4.62e−16 (20% of missing data), 9.91e−14 (30% of missing data), 1.77e−14 (40% of missing data), and 1.18e−12 (50% of missing data), respectively. Finally, the CD diagrams of the Nemenyi tests for the execution time are provided as online supporting information (see Figures S7–S11).

With respect to the average imputation error, GLSL had the best performance when the percentage of missing data was between 10% and 40%. The second position was disputed between MICE and HI-VAE. MICE obtained sligthly better results when the percentage of missing data was between 10% and 20%, and HI-VAE showed slightly better performance when the percentage of missing data was between 30% and 40%. HI-VAE improved its relative performance as the percentage of missing data increased, reaching first position at the 50% of

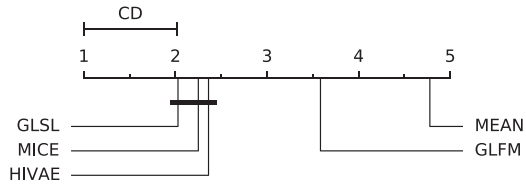

FIGURE 8 Critical difference diagram. Comparison of mean rank average imputation error with Nemenyi's post-hoc procedure when 20% of the data is missing. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; MEAN, mean imputation; MICE, multiple imputation chained equations

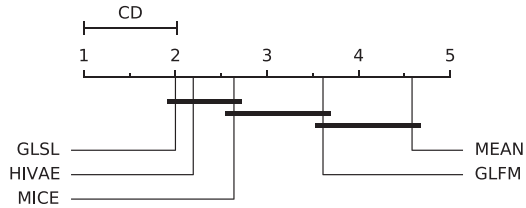

FIGURE 9 Critical difference diagram. Comparison of mean rank average imputation error with Nemenyi's post-hoc procedure when 30% of the data is missing. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; MEAN, mean imputation; MICE, multiple imputation chained equations

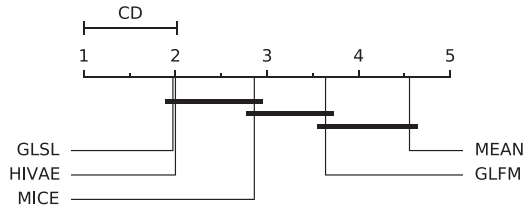

FIGURE 10 Critical difference diagram. Comparison of mean rank average imputation error with Nemenyi's post-hoc procedure when 40% of the data is missing. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; MEAN, mean imputation; MICE, multiple imputation chained equations

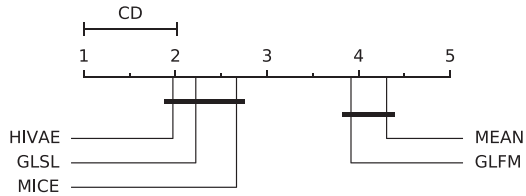

FIGURE 11 Critical difference diagram. Comparison of mean rank average imputation error with Nemenyi's post-hoc procedure when 50% of the data is missing. CD, critical difference; GLFM, general latent feature model; GLSL, greedy latent structure learner; MEAN, mean imputation; MICE, multiple imputation chained equations

missing data. No statistically significant differences were found between these methods. Nevertheless, both GLSL and HI-VAE performed significantly better than GLFM and MEAN in all of the missing data percentages.

With respect to the execution time, MEAN was the fastest method in all of the data sets, followed by MICE, which ended second in all of the data sets. The third position was disputed between GLFM and HI-VAE, and GLFM showed slightly faster results. GLSL was the slowest method, ending last in the majority of the data sets. Finally, execution times were practically unaffected by the percentage of missing data.

5 | CONCLUSIONS

In this paper, we developed an efficient multipartition clustering method, named GLSL, that is able to deal with real-world data sets that contain both categorical and continuous values. GLSL learns a CLG BN with multiple categorical latent variables, where each latent variable represents a unique partition of data. Since searching directly in the space of models would be computationally unfeasible, GLSL follows instead an iterative process with five latent operators and a VB version of the SEM algorithm. The main idea is to produce multiple candidate models that have introduced a new latent variable, removed an existing latent variable or changed the cardinality of a latent variable. Then, learn the local structure of the proposed candidates and select the refined candidate with the highest score.

We demonstrated the applicability of GLSL in a multipartition clustering task with PD data. Several partitions were identified, each of them grouping PD patients according to certain subsets of motor and/or nonmotor symptoms. Also, we used GLSL for density estimation with several real-world mixed data sets. We compared GLSL with several parametric and nonparametric methods. Although GLSL outperformed all of the considered methods, its implementation could be improved to obtain faster results.

Our proposal currently has several notable limitations. Some of them are intrinsic to the VB framework (and the VMP algorithm) while others are intrinsic to CLG BNs. All of these limitations could be addressed in the future to improve the performance and flexibility of our methods. First, we can extend our variational SEM by using more flexible variational families (instead of VB) and prior distributions,[75-77] but at the expense of a more difficult variational optimization problem. Second, we could consider a different divergence measure for the VI problem (other than the KL divergence). Several recent works have extended KL–VI to other statistical divergences such as Renyi's α-divergence,[78] Hellinger's α-divergence,[79] and χ-divergence.[80] Each of them has an associated bound that is used during optimization (similar to the ELBO in KL–VI). Furthermore, we could extend our work to f-VI,[81] which naturally unifies KL–VI, Rényi's α-VI, Hellinger's α-VI, and χ-VI. Third, we could extend our method to augmented CLG BNs, which allow categorical variables to have continuous parents by using the soft-max function.[82] Fourth, we could speed up GLSL by using parallelization (especially considering that multiple latent operators can be evaluated simultaneously). Finally, a future research direction is the investigation of a variant of variational autoencoders where there is a latent superstructure with multiple categorical latent variables[12,13] that is learned using GLSL.

ACKNOWLEDGMENTS

The authors would like to thank Carmen Rodriguez-Blazquez, Daniel Weintraub, Pablo Martinez-Martin, Alexandra Rizos, Anette Schrag, and Kallol Ray Chaudhuri for providing the

Parkinson's disease data. This study was supported in part by the Spanish Ministry of Economy and Competitiveness through the PID2019-109247GB-I00 project, by the European Union's Horizon 2020 Framework Programme for Research and Innovation under the Specific Grant Agreement No. 945539 (Human Brain Project SGA3), and by the BBVA Foundation (2019 Call) through the "Score-based nonstationary temporal Bayesian networks. Applications in climate and neuroscience" project.

ORCID

Fernando Rodriguez-Sanchez https://orcid.org/0000-0003-0790-6983
Concha Bielza https://orcid.org/0000-0001-7109-2668
Pedro Larrañaga https://orcid.org/0000-0002-1885-4501

REFERENCES

1. McLachlan GJ, Lee SX, Rathnayake SI. Finite mixture models. *Annu Rev Stat Appl*. 2019;6(1):355.
2. Zhang NL. Hierarchical latent class models for cluster analysis. *J Mach Learn Res*. 2004;5(6):697.
3. Galimberti G, Soffritti G. Model-based methods to identify multiple cluster structures in a data set. *Comput Stat Data Anal*. 2007;52(1):520.
4. Harmeling S, Williams CK. Greedy learning of binary latent trees. *IEEE Trans Pattern Anal Mach Intell*. 2011;33(6):1087.
5. Chen T, Zhang NL, Wang Y. Model-based multidimensional clustering of categorical data. *Artif Intell*. 2012;176(1):2246.
6. Liu TF, Zhang NL, Chen P, Liu AH, Poon LK, Wang Y. Greedy learning of latent tree models for multidimensional clustering. *Mach Learn*. 2015;98(1-2):301.
7. Rodriguez-Sanchez F, Larrañaga P, Bielza C. Discrete model-based clustering with overlapping subsets of attributes. In: *Proceedings of 9th International Conference on Probabilistic Graphical Models*. 2018:392.
8. Huang F, Naresh NU, Perros I, Chen R, Sun J, Anandkumar A. Guaranteed scalable learning of latent tree models. In: *Proceedings of 35th Conference on Uncertainty in Artificial Intelligence*; 2020:883.
9. Gu Y, Dunson DB. Identifying interpretable discrete latent structures from discrete data. (Preprint) arXiv:2101.10373, submitted: Jan 2021.
10. Poon LK, Zhang NL, Liu T, Liu AH. Model-based clustering of high-dimensional data: variable selection versus facet determination. *Int J Approx Reason*. 2013;54(1):196.
11. Galimberti G, Manisi A, Soffritti G. Modelling the role of variables in model-based cluster analysis. *Stat Comput*. 2018;28(1):145.
12. Li X, Chen Z, Poon LK, Zhang NL. Learning latent superstructures in variational autoencoders for deep multidimensional clustering. In: *Proceedings of 7th International Conference on Learning Representations*. 2019:1.
13. Falck F, Zhang H, Willetts M, Nicholson G, Yau C, Holmes CC. Multif-facet clustering variational autoencoders. (Preprint) arXiv:2106.05241, submitted: Jan 2021.
14. Vanderwalle V. Multi-partitions subspace clustering. *Mathematics*. 2020;8(4):597.
15. Marbac M, Vanderwalle V. A tractable multi-partitions clustering. *Comput Stat Data Anal*. 2019;132:167.
16. Zhou C, Wang X, Guo J. Learning mixed latent tree models. *J Mach Learn Res*. 2020;21(249):1.
17. McNicholas PD. Model-based clustering. *J Classif*. 2016;33(3):331.
18. Pearl J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann; 1988.
19. Koller D, Friedman N. *Probabilistic Graphical Models: Principles and Techniques*. The MIT Press; 2009.
20. Lauritzen SL, Wermuth N. Graphical models for associations between variables, some of which are qualitative and some quantitative. *Ann Stat*. 1989;17:31.
21. Scanagatta M, Salmerón A, Stella F. A survey on Bayesian network structure learning from data. *Prog Artif Intell*. 2019;8:425.
22. Akaike H. A new look at the statistical model identification. *IEEE Trans. Automat Contr*. 1974;19(6):716.
23. Schwarz G. Estimating the dimension of a model. *Ann Stat*. 1978;6(2):461.

24. Friedman N. Learning belief networks in the presence of missing values and hidden variables. In: *Proceedings of 14th International Conference on Machine Learning*. 1997:125.
25. Dempster AP, Laird NM, Rubin DB. Maximum likelihood from incomplete data via the EM algorithm. *J R Stat Soc (Series B Stat Methodol)*. 1977;39(1):1.
26. Attias H. A variational Bayesian framework for graphical models. In: *Proceedings of 14th Conference on Neural Information Processing Systems*. 2000:209.
27. Caruana R, Elhawary M, Nguyen N, Smith C. Meta clustering. In: *Proceedings of 6th IEEE International Conference on Data Mining*. 2006:107.
28. Niu D, Ghahramani Z. A nonparametric Bayesian model for multiple clusterings with overlapping feature views. In: *Proceedings of 15th International Conference on Artificial Intelligence and Statistics*. 2012:814.
29. Ye W, Maurus S, Hubig N, Plant C. Generalized independent subspace clustering. In: *Proceedings of 16th IEEE International Conference on Data Mining*. 2016:569.
30. Mautz D, Ye W, Plant C, Böhm C. Non-redundant subspace clusterings with Nr-Kmeans and Nr-DipMeans. *ACM Trans Knowl Discov Data*. 2020;14(5):1.
31. Dang XH, Bailey J. A framework to uncover multiple alternative clusterings. *Mach Learn*. 2015;98:7.
32. Yang S, Zhang L. Non-redundant multiple clustering by nonnegative matrix factorization. *Mach Learn*. 2017;106(5):695.
33. Hu J, Qian Q, Pei J, Jin RM, Zhu S. Finding multiple stable clusterings. *Knowl Inf Syst*. 2017;51(3):991.
34. Kriegel HP, Zimek A. Subspace clustering, ensemble clustering, alternative clustering, multiview clustering: what can we learn from each other. In: *Proceedings of 1st International Workshop on Discovering, Summarizing and Using Multiple Clusterings*. 2010:9.
35. Bickel S, Scheffer T. Multi-view clustering. In: *Proceedings of 4th IEEE International Conference on Data Mining*. 2004:19.
36. Yang Y, Wang H. Multi-view clustering: a survey. *Big Data Min Anal*. 2018;1(2):83.
37. Dong X, Yu Z, Cao W, Shi Y, Ma Q. A survey on ensemble learning. *Front Comput Sci*. 2020;14(2):241.
38. Tang C, Zheng X, Liu X, Zhang W, Zhang J, Xiong J, Wang L. Cross-view locality preserved diversity and consensus learning for multi-view unsupervised feature selection. *IEEE Trans. Knowl. Data Eng*. 2021. doi:10.1109/TKDE.2020.3048678
39. Lerner U, Parr R. Inference in hybrid networks: theoretical limits and practical algorithms. In: *Proceedings of 17th Conference on Uncertainty in Artificial Intelligence*. 2001:310.
40. Hastings WK. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*. 1970;57(1):97.
41. Blei DM, Kucukelbir A, McAuliffe JD. Variational inference: a review for statisticians. *J Am Stat Assoc*. 2016;112(518):859.
42. Winn J, Bishop CM. Variational message passing. *J Mach Learn Res*. 2005;6:661.
43. Chickering D, Geiger D, Heckerman D. Learning Bayesian networks: search methods and experimental results. In: *Proceedings of 5th International Workshop on Artificial Intelligence and Statistics*. 1995:112.
44. Beal MJ, Ghahramani Z. The variational Bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian Stat*. 2003;7(210):453.
45. Benjumeda M, Luego-Sanchez S, Larrañaga P, Bielza C. Tractable learning of Bayesian networks from partially observed data. *Pattern Recognit*. 2019;91:190.
46. Kwisthout J, Boadlaender HL, van der Gaag LC. The necessity of bounded treewidth for efficient inference in Bayesian networks. In: *Proceedings of 19th European Conference on Artificial Intelligence*. 2010:237-242.
47. Bishop CM. *Pattern Recognition and Machine Learning*. Springer; 2016.
48. Casella G. An introduction to empirical Bayes data analysis. *Am. Stat*. 1985;39(2):83.
49. Kwisthout K. Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artif Intel*. 1993;60(1):141.
50. Rodriguez-Sanchez F, Larrañaga P, Bielza C. Incremental learning of latent forests. *IEEE Access*. 2020;8:224420.
51. Masegosa AR, Martínez AM, Ramos-López D, et al. AMIDST: a Java toolbox for scalable probabilistic machine learning. *Knowl Based Syst*. 2019;163:595.
52. Chaudhuri KR, Schrag A, Weintraub D, et al. The movement disorder society nonmotor rating scale: initial validation study. *Mov Disord*. 2019;35(1):116.

53. Goetz CG, Tilley BC, Shaftman SR, et al. Movement disorder society-sponsored revision of the unified parkinson's disease rating scale (MDS-UPDRS): scale presentation and clinimetric testing results: mds-updrs: clinimetric assessment. *Mov Disord*. 2008;23(15):2129.
54. Chaudhuri KR, Martinez-Martin P, Brown RG, et al. The metric properties of a novel non-motor symptoms scale for Parkinson's disease: results from an international pilot study. *Mov Disord*. 2007;22(13):1901.
55. Mu J, Chaudhuri KR, Bielza C, Pedro-Cuesta J, Larrañaga P, Martinez-Martin P. Parkinson's disease subtypes identified from cluster analysis of motor and non-motor symptoms. *Front Aging Neurosci*. 2017;9:301.
56. Marras C, Chaudhuri KR, Titova N, Mestre TA. Therapy of Parkinsonâ's disease aubtypes. *Neurotherapeutics*. 2020;17(4):1366.
57. Mann HB, Whitney DR. On a test of whether one of two random variables is stochastically larger than the other. *Ann Math Stat*. 1947:50.
58. Kruskal WH, Wallis WA. Use of ranks in one-criterion variance analysis. *J Am Stat Assoc*. 1952;47(260):583.
59. Tukey J. Comparing individual means in the analysis of variance. *Biometrics*. 1949;5(2):99.
60. Lazarsfeld PF, Henry NW. *Latent Structure Analysis*. Houghton Mifflin; 1968.
61. Pham DT, Ruz GA. Unsupervised training of Bayesian networks for data clustering. *Proc R Soc Lond A Math Phys Sci*. 2009;465(2109):2927.
62. McParland D, Gormley IC. Model based clustering for mixed data: clustMD. *Adv Data Anal Classif*. 2016;10(2):155.
63. Marbac M, Biernacki C, Vanderwalle V. Model-based clustering of Gaussian copulas for mixed data. *Commun Stat Theory Methods*. 2017;46(23):11635.
64. Dua D, Graff C. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml
65. Friedman M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *J Am Stat Assoc*. 1937;32:675.
66. Nemenyi PB. Distribution-free multiple comparisons. PhD Thesis, Princeton University; 1963.
67. Li Q, Racine J. Nonparametric estimation of distributions with categorical and continuous data. *J Multivar Anal*. 2003;86(2):266.
68. Molina A, Vergari A, DiMauro N, Natarajan S, Esposito F, Kersting K. Mixed sum-product networks: a deep architecture for hybrid domains. In: *Proceedings 32nd AAAI Conference on Artificial Intelligence*. 2018:3828.
69. Poon H, Domingos P. Sum-product networks: a new deep architecture. In: *Proceedings of 27th Conference on Uncertainty in Artificial Intelligence*. 2011:337.
70. Valera I, Pradier MF, Lomeli M, Ghahramani Z. General latent feature models for heterogeneous datasets. *J Mach Learn Res*. 2020;21(100):1.
71. Ma C, Tschiatschek S, Turner R, Hernńndez-Lobato JM, Zhang C. VAEM: a deep generative model for heterogeneous mixed type data. In: *Proceedings 34th Conference on Neural Information Processing Systems*. 2020:11237.
72. Demšar J. Statistical comparisons of classifiers over multiple data sets. *J Mach Learn Res*. 2006;7(1):1.
73. Azur MJ, Stuart EA, Frangakis C, Leaf PJ. Multiple imputation by chained equations: what is it and how does it work?: multiple imputation by chained equations. *Int J Methods Psychiatr Res*. 2011;20(1):40.
74. Nazabal A, Olmos PM, Ghahramani Z, Valera I. Handling incomplete heterogeneous data using VAEs. *Pattern Recognit*. 2020;107(11):107501.
75. Bishop CM, Lawrence ND, Jaakola T, Jordan MI. Approximating posterior distributions in belief networks using mixtures. In: *Proceedings of 12th Conference on Neural Information Processing Systems*. 1998:416.
76. Barber D, Wiegerinck W. Tractable variational structures for approximating graphical models. In: *Proceedings of 13th Conference on Neural Information Processing Systems*. 1999:183.
77. Wang C, Blei DM. Variational inference in nonconjugate models. *J. Mach. Learn. Res*. 2013;14(4):1005.
78. Li Y, Turner RE. Rényi divergence variational inference. In: *Proceedings of 30th Conference on Neural Information Processing Systems*. 2016:1081.
79. Sason I. On f-divergences: integral representations, local behavior, and inequalities. *Entropy*. 2018;20(5):383.
80. Dieng AB, Tran D, Ranganath R, Paisley J, Blei DM. Variational inference via chi upper bound minimization. In: *Proceedings of 31th Conference on Neural Information Processing Systems*. 2017:2729.

81. Wan N, Li D, Hovakimyan N. f-divergence variational inference. In: *Proceedings of 34th Conference on Neural Information Processing Systems*. 2020:17370.
82. Koller D, Lerner U, Angelov D. A general algorithm for approximate inference and its aplication to hybrid Bayes nets. In: *Proceedings of 15th Conference on Uncertainty in Artificial Intelligence*. 1999:324.

SUPPORTING INFORMATION

Additional supporting information may be found in the online version of the article at the publisher's website.

How to cite this article: Rodriguez-Sanchez F, Bielza C, Larrañaga P. Multipartition clustering of mixed data with Bayesian networks. *Int J Intell Syst*. 2022;37:2188-2218. doi:10.1002/int.22770