Classification and clustering with continuous time Bayesian network models

Article *in* Journal of Intelligent Information Systems · November 2014

DOI: 10.1007/s10844-014-0345-0

2 authors, including:

Fabio Stella
Università degli Studi di Milano-Bicocca
71 PUBLICATIONS **425** CITATIONS

SEE PROFILE

Journal of Intelligent Information System manuscript No.
(will be inserted by the editor)

Classification and Clustering with Continuous Time Bayesian Network Models

Daniele Codecasa · Fabio Stella

Received: date / Accepted: date

Abstract Classification and clustering of streaming data are relevant in finance, computer science, and engineering while they are becoming increasingly important in medicine and biology. Streaming data are analyzed with algorithms and models capable to represent dynamics, sequences and time. Dynamic Bayesian networks and hidden Markov models are commonly used to analyze streaming data. However, they are concerned with evenly spaced time series data and thus suffer from several limitations. Indeed, it is not clear how timestamps should be discretized even if some approaches to mitigate this problem have been recently made available. In this paper we describe the class of continuous time Bayesian networks classifiers and develop algorithms for their parametric and structural learning to solve classification and clustering of multivariate discrete state continuous time trajectories. Numerical experiments on synthetic and real world data are used to compare the performance of continuous time Bayesian network models to that achieved by dynamic Bayesian networks. In particular, post-stroke rehabilitation data is used for the classification task while urban traffic data from continuous time loop is used for the clusteirng task. The achieved results confirm the effectiveness of the proposed approaches.

Keywords streaming data · multivariate trajectory · continuous time classification · continuous time clustering · continuous time Bayesian networks

Daniele Codecasa
DISCo, Università degli Studi di Milano-Bicocca,
Viale Sarca 336, 20126 Milano, Italy
E-mail: codecasa@disco.unimib.it

Fabio Stella
DISCo, Università degli Studi di Milano-Bicocca,
Viale Sarca 336, 20126 Milano, Italy
E-mail: stella@disco.unimib.it

1 Introduction

Streaming data are relevant in finance, computer science, and engineering while they are becoming increasingly important in medicine and biology [5]. High frequency trading is an instance where streaming data is relevant to finance [13,72]. Computer science offers many examples of streaming data, system error logs, web search query logs, network intrusion detection and social networks to mention just a few [64]. Image, acoustic and vision processing are applications where streaming data are used to solve engineering problems [78]. An emerging paradigm in medicine is that of patient monitoring and continuous time diagnosis based on sensor data, including the study of computational firing patterns of neurons [71]. Finally, the increment of the amount of time course data generated in biology allows to discover gene regulatory networks [1], to model the evolution of infections and to learn and analyze metabolic networks [73].

Data streaming problems may be approached with algorithms and models capable of representing dynamics, sequences and time. Among these, dynamic Bayesian networks [15] and hidden Markov models [55] received great attention for modeling temporal dependencies. However, dynamic Bayesian networks are concerned with discrete time and thus suffer from several limitations due to the fact that it is not clear how timestamps should be discretized. In the case where a too slow sampling rate is used, the data will be poorly represented; while a too fast sampling rate rapidly makes learning and inference prohibitive. Furthermore, it has been pointed out [33] that when allowing long term dependencies it is required to condition on multiple steps into the past, and thus choosing a too fast sampling rate will increase the number of such steps that need to be conditioned on. Recently an interesting method to convert dynamic Bayesian networks with a naturally small time steps to ones with a larger time step has been developed [7]. This method requires a fixed step size and thus it does not address the case where events occurring between the chosen time step intervals are important as well as the case where the dynamics of the model are not uniform over time [19]. However, it is worthwhile to mention that DBNs provide in many cases a reasonable approximation to the data generating process.

Continuous time Bayesian networks [47], continuous time noisy-or (CT-NOR) [63], Poisson cascades [64], Poisson networks [56], piecewise-constant conditional intensity model (PCIM) [33], together with forest-based point processes [75] are interesting models to represent and analyze continuous time processes. CT-NOR and Poisson cascades model event streams, while they require the modeler to specify a parametric form for temporal dependencies. This aspect significantly impacts performance, and the problem of model selection in CT-NOR and Poisson cascades has not been addressed yet. This limitation has been overcome by PCIMs which perform structure learning to model how events in the past affect future events of interest. Continuous time Bayesian networks are homogeneous Markov models which allow to represents

joint trajectories of discrete, finite domain variables, rather than models of event streams in continuous time.

In this paper continuous time Bayesian network models are developed for solving classification [67] and clustering problems of multivariate temporal trajectories. Max-k continuous time Bayesian networks classifier and max-k augmented continuous time naive Bayes are defined. An algorithm which optimizes a conditional log-likelihood scoring function to learn continuous time Bayesian network classifiers [9] is presented. New contributions concerning parametric and structural learning for continuous time Bayesian network models are developed to solve the problem of clustering multivariate discrete state continuous time trajectories. Synthetic and real world data are used to compare the classification and clustering performance achieved by continuous time Bayesian network models to that achieved by dynamic Bayesian network models. The present work extends [9] by analyzing and developing for the first time continuous time Bayesian network models to solve the clustering problem on discrete state continuous time trajectory data. Therefore, in this paper, the comparison of the classification performance is presented only for the post-stroke rehabilitation problem, while a detailed performance comparison on synthetic data is reported in [11]. Performance comparison for the clustering task is reported for synthetic and real world data. In particular, the performance of continuous time Bayesian network models is analyzed in the case where multivariate temporal trajectory data is used to solve the complex problem of learning urban traffic profiles from loop data. The achieved results confirm the effectiveness of the proposed approach and offer an innovative solution for recognizing urban traffic profiles.

The rest of the paper is organized as follows. Section 2 is devoted to related works. Basic definitions and notations concerning continuous time Bayesian networks are given in Section 3. Section 4 is devoted to present continuous time Bayesian network classifiers while Section 5 describes an algorithm for clustering continuous time multivariate trajectory data. Section 6 presents numerical experiments for both the classification and clustering tasks. In this section the developed classification and clustering models are used to solve two complex real world problems, namely post-stroke rehabilitation and urban traffic profiling. Both classification and clustering numerical experiments on continuous time Bayesian network classifiers are performed using the CTBNCToolkit [10]. Conclusions and further research directions are described in Sections 7 and 8.

2 Related works

2.1 Time stream data analysis

Classification and clustering have been studied and developed in the stream analysis research area. In [16,38] the authors developed approaches based on decision trees to solve the problem of supervised classification of streaming data. The *on-demand classification algorithm* has been introduced in [2] to

allow supervised classification of streaming data when collecting statistics over time. In [27] the authors propose a lightweight classification technique, while an adaptive nearest classifier has been introduced in [44].

The specialized literature concerning unsupervised classification consists of many different approaches to deal with data streams [62]. In [6] the authors propose a variation of the k-means algorithm and apply it to very large data sets, while [21] presents a scalable k-means algorithm to improve computational efficiency. ClusTree algorithm [41] has been introduced to deal with continuously incoming data with possibly varying inter-arrival times while the SWClustering algorithm [80] tracks the evolution of clusters over sliding windows capturing temporal cluster features.

Continuous time Bayesian networks (CTBNs) are probabilistic graphical models introduced by Nodelman et al [48] to overcome the limitations of dynamic Bayesian networks through explicit modelling of the temporal dynamics. In [67] continuous time Bayesian network classifiers are introduced as a specialization of CTBNs for the classification purpose of a variable which does not change in time. In [9] and [11] the structural learning algorithm for CTBNCs using marginal log-likelihood and conditional log-likelihood scores is introduced and described. This paper extends the work in [9] to study for the first time the application of continuous time Bayesian network models to solve the problem of clustering discrete state continuous time trajectory data.

2.2 Post stroke rehabilitation

Rehabilitation after clinical treatment is an important challenge for health systems. Tormene et al [70] pointed out how this is also true for post-stroke rehabilitation. In term of costs and effectiveness it is very important to start physical therapy as soon as possible [52, 42, 24]. Tormene et al [70] proposed an automatic movement recognition system to face these difficulties. The idea is to provide the patient with a system capable of recognizing the movements and to inform him/her about their correctness. Movement correctness must be assessed in real time before the exercise is completed and before the full trajectory becomes available.

The authors focused on the upper limb post-stroke rehabilitation and analyzed seven rehabilitation exercises. For each exercise 120 multivariate trajectories were collected, where the values of 29 sensors are recorded with a frequency of 30 Hz. Each movement is addressed separately as a classification problem. Tormene et al [70] used the 1-Nearest Neighbor Classifier algorithm (NNC) applied to dynamic time warping (DTW) [39] and to open end DTW (OE-DTW) distances.

In Section 6.1 continuous time Bayesian network classifiers are compared to dynamic Bayesian networks and with the results described in Tormene et al [70].

2.3 Urban traffic profile

Traffic congestion is one of the main concerns to cope with in all the big cities of the world. Wasting time driving a car is source of stress and has big impacts on economic activities. In [4] the authors estimate a cost of 48 billion dollars due to traffic congestion in 39 metropolitan areas in the United States with a population of one million or more. This does not include the cost due to unpredictability of traffic delays, the cost of extra fuel and air pollution. The environmental issue is not less important. Traffic is one of the main causes of environmental and acoustic pollution which leads to high social costs. The relationships between pollution and allergies, asthma, tumors and, immune system diseases are clear [32,45]. The analysis of these data lead to the conclusion that traffic is an major social problem that must be tackled.

Urban traffic control (UTC) is one of the major challenges for traffic engineers. It promises to be one of the most effective ways to cope with traffic congestion in metropolitan areas. This is particularly true when considering the evolution of new technologies, such as sensors to monitor the streets (i.e. loops, cameras, ...) together with the increased computational capability of modern computers. Nevertheless, because traffic is a chaotic system, and traffic profile changes continuously during the day, the traffic light control is still an open problem. Traffic light control and coordination is a well studied problem in the specialized literature [53]. The following commercial solutions are very important: TRANSYT [59], an off-line optimization model; SCOOT [60], SCATS [65] and UTOPIA[1] which offer traffic-responsive strategies.

The improvement of techniques from artificial intelligenge has originated new approaches to model and optimize complex transportation system [68]. The specialized literature describes many instances where the traffic light control problem is tackled with artificial intelligence techniques and models. In [79] the authors propose a discrete-time, stationary, Markov decision process in order to solve the problem of traffic control. In [34] has been proposed a Markov decision process decomposition approach to control isolated intersections. A reinforcement learning approach is used in [69] and [76] to define a UTC system with learning capability. Also approaches based on expert systems [23,37], fuzzy logic [22,3], neural networks [66], auto-organization systems [29,43] and evolutionary algorithms [54] were proposed. However, in practice only few of these approaches can be applied to real world situations. This is because of their need to make strong assumptions and ineffective approximations to address the computational effort of managing real networks and complex systems. Clustering of the traffic profiles (i.e. the state of the traffic) is a way to simplify the complex problem of traffic light optimization. The idea is to cluster the traffic condition off-line in order to find the best plan for each particular traffic condition. Then, in real time the off-line optimized traffic light control plan associated with the urban traffic profile which is the most similar to the current traffic condition is used. Eventually some modifications are applied

[1] http://www.swarco.net/

to the off-line optimal plan to adapt it to the current traffic condition. This approach is not new in the literature: in [3] the authors use fuzzy clustering of origin-destination matrix as an intermediate step to their urban traffic control approach.

In Section 6.2.4 the problem of clustering traffic profiles is addressed by comparing the performance of continuous time Bayesian network models to that of dynamic Bayesian networks.

3 Continuous Time Bayesian Networks

3.1 Basic definitions

Dynamic Bayesian networks (DBNs) model dynamic systems without representing time explicitly. They discretize time to represent a dynamical system through several time slices. However, Nodelman in [48] pointed out that "*since DBNs slice time into fixed increments, one must always propagate the joint distribution over the variables at the same rate*". Therefore, if the system consists of processes which evolve at different time granularities and/or the obtained observations are irregularly spaced in time, the inference process may become computationally intractable.

Continuous time Bayesian networks (CTBNs) overcome the limitations of DBNs by explicitly representing temporal dynamics. Therefore, they allow us to recover the probability distribution over time when specific events occur. CTBNs are based on *homogeneous Markov processes* where transition intensities do not depend on time.

A continuous time Bayesian network (CTBN) is a probabilistic graphical model whose nodes are associated with random variables and whose state evolves in continuous time. Evolution of each variable depends on the state of its parents in the graph associated with the CTBN model.

Definition 1 (Continuous time Bayesian network). [48]. Let $\mathbf{X}$ be a set of random variables $X_1, X_2, ..., X_N$. Each X_n has a finite domain of values $Val(X_n) = \{x_1, x_2, ..., x_{I_n}\}$. A continuous time Bayesian network $\aleph$ over $\mathbf{X}$ consists of two components: the first is an initial distribution $P^0_{\mathbf{X}}$, specified as a Bayesian network $\mathcal{B}$ over $\mathbf{X}$. The second is a continuous transition model, specified as:

- a directed (possibly cyclic) graph $\mathcal{G}$ whose nodes are $X_1, X_2, ..., X_N$; $Pa(X_n)$ denotes the parents of X_n in $\mathcal{G}$.
- a conditional intensity matrix, $\mathbf{Q}_{X_n}^{Pa(X_n)}$, for each variable $X_n \in \mathbf{X}$.

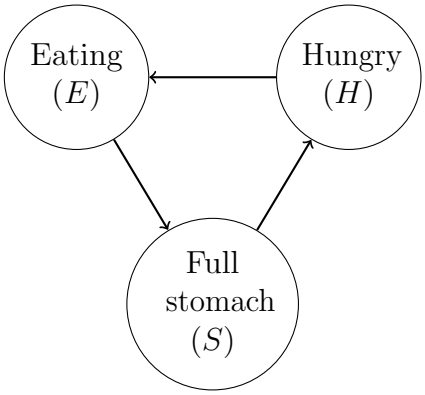


Fig. 1 A part of the drug network.

Given the random variable X_n, the *conditional intensity matrix* (CIM) $\mathbf{Q}_{X_n}^{Pa(X_n)}$ consists of a set of intensity matrices, one intensity matrix

$$\mathbf{Q}_{X_n}^{pa(X_n)} = \begin{bmatrix} -q_{x_1}^{pa(X_n)} & q_{x_1x_2}^{pa(X_n)} & . & q_{x_1x_{I_n}}^{pa(X_n)} \\ q_{x_2x_1}^{pa(X_n)} & -q_{x_2}^{pa(X_n)} & . & q_{x_2x_{I_n}}^{pa(X_n)} \\ . & . & . & . \\ q_{x_{I_n}x_1}^{pa(X_n)} & q_{x_{I_n}x_2}^{pa(X_n)} & . & -q_{x_{I_n}}^{pa(X_n)} \end{bmatrix},$$

for each instantiation $pa(X_n)$ of the parents $Pa(X_n)$ of node X_n, where $q_{x_i}^{pa(X_n)} = \sum_{j \neq i} q_{x_ix_j}^{pa(X_n)}$ is the rate of leaving state x_i for a specific instantiation $pa(X_n)$ of $Pa(X_n)$, while $q_{x_ix_j}^{pa(X_n)}$ is the rate of arriving to state x_j from state x_i for a specific instantiation $pa(X_n)$ of $Pa(X_n)$. Matrix $\mathbf{Q}_{X_n}^{pa(X_n)}$ can equivalently be summarized by using two types of parameters, $q_{x_i}^{pa(X_n)}$ that is associated with each state x_i of the variable X_n when its parents are set to $pa(X_n)$, and $\theta_{x_ix_j}^{pa(X_n)} = \frac{q_{x_ix_j}^{pa(X_n)}}{q_{x_i}^{pa(X_n)}}$ which represents the probability that the variable X_n transitions from state x_i to state x_j, when it is known that the transition occurs at a given instant in time and the parents are set to $pa(X_n)$.

Example 1 Figure 1 shows a part of the drug network introduced in [48]. It contains a cycle, that indicates whether a person is hungry (H) depending on how full his/her stomach (S) is, which depends on whether or not he/she is eating (E), which in turn depends on whether he/she is hungry. We assume that E and H are binary variables with states *no* and *yes* while the variable S can be in one of the following states; *full*, *average* or *empty*. Then, the variable E is fully specified by the [2×2] CIM matrices $\mathbf{Q}_E^{no}$, and $\mathbf{Q}_E^{yes}$, the variable S is fully specified by the [3×3] CIM matrices $\mathbf{Q}_S^{no}$ and $\mathbf{Q}_S^{yes}$, while the the variable H is fully specified by the [2×2] CIM matrices $\mathbf{Q}_H^{ful}$, $\mathbf{Q}_H^{ave}$ and $\mathbf{Q}_H^{emp}$. For matters of brevity we only show $\mathbf{Q}_S^{yes}$ with two equivalent parametric

representations:

$$\mathbf{Q}_S^{yes} = \begin{bmatrix} -q_{ful}^{yes} & q_{ful,ave}^{yes} & q_{ful,emp}^{yes} \\ q_{ave,ful}^{yes} & -q_{ave}^{yes} & q_{ave,emp}^{yes} \\ q_{emp,ful}^{yes} & q_{emp,ave}^{yes} & -q_{emp}^{yes} \end{bmatrix} = \begin{bmatrix} -0.03 & 0.02 & 0.01 \\ 5.99 & -6.00 & 0.01 \\ 1.00 & 5.00 & -6.00 \end{bmatrix} \tag{1}$$

$$\begin{aligned} \mathbf{Q}_S^{yes} &= \begin{bmatrix} q_{ful}^{yes} & 0 & 0 \\ 0 & q_{ave}^{yes} & 0 \\ 0 & 0 & q_{emp}^{yes} \end{bmatrix} \left(\begin{bmatrix} 0 & \theta_{ful,ave}^{yes} & \theta_{ful,emp}^{yes} \\ \theta_{ave,ful}^{yes} & 0 & \theta_{ave,emp}^{yes} \\ \theta_{emp,ful}^{yes} & \theta_{emp,ave}^{yes} & 0 \end{bmatrix} - \mathbf{I} \right) \\ &= \begin{bmatrix} 0.03 & 0.00 & 0.00 \\ 0.00 & 6.00 & 0.00 \\ 0.00 & 0.00 & 6.00 \end{bmatrix} \left(\begin{bmatrix} 0 & \frac{0.02}{0.03} & \frac{0.01}{0.03} \\ \frac{5.99}{6.00} & 0 & \frac{0.01}{6.00} \\ \frac{1.00}{6.00} & \frac{5.00}{6.00} & 0 \end{bmatrix} - \mathbf{I} \right) \end{aligned} \tag{2}$$

where $\mathbf{I}$ is the identity matrix.

If we view units of time as hours, then we expect a person who has an empty stomach (S=*empty*) and is eating (E=*yes*) to stop having an empty stomach in 10 minutes ($\frac{1}{6}$ hour). The stomach will then transition from state *empty* (S=*empty*) to state *average* (S=*average*) with probability $\frac{5}{6}$ and to state $full$ (S=*full*) with probability $\frac{1}{6}$. Equation (1) is a compact representation of the CIM while Equation (2) is useful because it explicitly represents the transition probability value from state x to state x', i.e. $\theta_{xx'}^{pa(X)}$.

CTBNs allow two types of evidence, namely *point evidence* and *continuous evidence*, while HMMs and DBNs allow only point evidence. *Point evidence* at time t for a subset of variables $X_1, X_2, ..., X_k \in \mathbf{X}$ is the knowledge of the states $x_1, x_2, ..., x_k$ at time t for the variables $X_1, X_2, ..., X_k$. Point evidence will be referred to as $X_1^t = x_1^t, X_2^t = x_2^t, ..., X_k^t = x_k^t$ or in compact notation as $\mathbf{X}^t = \mathbf{x}^t$, where $\mathbf{X}^t = (X_1^t, X_2^t, ..., X_k^t)$ while $\mathbf{x}^t = (x_1^t, x_2^t, ..., x_k^t)$. *Continuous evidence* is the knowledge of the states $x_1, x_2, ..., x_k$ of a set of variables $X_1, X_2, ..., X_k \in \mathbf{X}$ throughout an entire interval of time $[t_1, t_2)$ (that it is taken to be a half-closed interval).

3.2 Inference

A CTBN exploits conditional independence relationships between variables to obtain a factored representation of a homogeneous Markov process. Given the CIMs associated with the variables of the CTBN $\aleph$, the *amalgamation* operation [48] over the CIMs allows to recover the *joint intensity matrix* of a homogenous Markov process.

$$\mathbf{Q}_\aleph = \prod_{X_n \in \mathbf{X}} \mathbf{Q}_{X_n}^{Pa(X_n)}. \tag{3}$$

Given the joint intensity matrix $\mathbf{Q}_\aleph$ (3) it is possible to compute the joint distribution of the variables $X_1, X_2, ..., X_N$ as follows [48]:

$$P_\aleph(t) = P_\aleph^0 \exp(\mathbf{Q}_\aleph\, t),$$

where $P^0_{\aleph}$ is the initial distribution of the variables $X_1, X_2, ..., X_N$, compactly represented as a Bayesian network $\mathcal{B}$ in the CTBN framework. Similarly the joint distribution over two time points s and t can be computed as follows [48]:

$$P_{\aleph}(s,t) = P_{\aleph}(s) \exp\left(\mathbf{Q}_{\aleph}\,(t-s)\right),$$

with $t > s$. The above formula shows that CTBNs and homogenous Markov processes inherit the memoryless property from the exponential distribution.

Inference in CTBNs can be performed by exact and approximate algorithms. *Full amalgamation* [48] is an exact algorithm that involves generating the exponentially-large matrix $\mathbf{Q}_{\aleph}$ (3) representing the transition model over the entire state space. Exact inference in CTBNs is known to be intractable [48, 47], and thus different approximate algorithms have been proposed. Nodelman at al [47] introduced the *Expectation Propagation* (EP) algorithm which allows both point and interval evidence. In [61] the authors presented a new EP-based algorithm which uses a flexible cluster graph architecture that fully exploits the natural time-granularity at which different sub-processes evolve. Alternatives are offered by the continuous time belief propagation [17], mean field variational approximation [12] and sampling based inference algorithms, such as importance sampling [20] and *Gibbs sampling* [18, 58].

3.3 Parameter learning

Given a data set $\mathcal{D}$ and a fixed structure of a CTBN, parameter learning is based on *marginal log-likelihood estimation*, and takes into account the *imaginary counts* of the hyperparameters $\alpha_x^{pa(X)}$, $\alpha_{xx'}^{pa(X)}$ and, $\tau_x^{pa(X)}$. The parameters $q_x^{pa(X)}$ and $\theta_{xx'}^{pa(X_n)}$ can be estimated as follows:

- $q_x^{pa(X)} = \frac{\alpha_x^{pa(X)} + M[x|pa(X)]}{\tau_x^{pa(X)} + T[x|pa(X)]}$;
- $\theta_{xx'}^{pa(X)} = \frac{\alpha_{xx'}^{pa(X)} + M[x,x'|pa(X)]}{\alpha_x^{pa(X)} + M[x|pa(X)]}$,

where

- $M[x, x' \mid pa(X)]$: number of times X transitions from state x to state x' when the state of its parents $Pa(X)$ is set to $pa(X)$;
- $M[x \mid pa(X)] = \sum_{x' \neq x} M[x, x' \mid pa(X)]$: number of times X leaves the state x when the state of its parents $Pa(X)$ is set to $pa(X)$;
- $T[x \mid pa(X)]$: amount of time X spends in state x when the state of its parents $Pa(X)$ is set to $pa(X)$,

are *sufficient statistics* computed over the data set $\mathcal{D}$.

3.4 Structural learning

Learning the structure of a CTBN from a given data set $\mathcal{D}$ has been addressed as an optimization problem over possible CTBN structures [49]. It consists of

finding the structure $\mathcal{G}^*$ which maximizes the following Bayesian score:

$$\mathbf{score}_{\aleph}(\mathcal{G} : \mathcal{D}) = \ln P(\mathcal{D}|\mathcal{G}) + \ln P(\mathcal{G}). \quad (4)$$

However, the search space of this optimization problem is significantly simpler than that of BNs and DBNs. All edges are across time and thus represent the effect of the current value of one variable on the next value of the other variables. Therefore, no acyclicity constraints arise, and it is possible to optimize the parent set for each variable of the CTBN independently. It has been shown that learning the optimal structure of a BN is NP-hard [8]. On the contrary, learning the optimal structure of a CTBN, once fixed the maximum number of parents, is polynomial with respect to the number of variables and the dimension of the data set [49].

4 CTBNs for Classification

4.1 Introduction

Continuous time Bayesian network classifiers (CTBNCs) [67] are a specialization of CTBNs for the classification of a static variable. They allow polynomial time classification, while inference on general CTBNs is NP-hard. Classifiers from this class explicitly represent the evolution in continuous time of the joint state of a set of random variables X_n, $n = 1, 2, ..., N$ which are assumed to depend on the class node Y whose state does not change over time.

Definition 2 (Continuous time Bayesian network classifier)[2]. A continuous time Bayesian network classifier is a pair $\mathcal{C} = \{\aleph, P(Y)\}$ where $\aleph$ is a CTBN model with attribute nodes $X_1, X_2, ..., X_N$, Y is the class node with marginal probability $P(Y)$ on states $Val(Y) = \{y_1, y_2, ..., y_S\}$, $\mathcal{G}$ is the graph of the CTBNC, such that the following conditions hold:

- $Pa(Y) = \emptyset$, the class variable Y is associated with a root node;
- Y is fully specified by $P(Y)$ and does not depend on time.

An instance of a CTBNC consisting of six attributes $X_1, X_2, ..., X_6$ and the class Y is depicted in Figure 2. The class variable Y is associated with the root node. It is worthwhile to notice that the model contains a cycle involving nodes X_3 and X_4, which is allowed in CTBNs while not in BNs. .

Given a data set $\mathcal{D}$ with no missing data, a CTBNC can be learned by maximizing the Bayesian score function $\mathbf{score}_{\aleph}(\mathcal{G} : \mathcal{D})$ (4) subjected to the constraints listed in Definition 2. Exact learning requires to set in advance the maximum number of parents k for the nodes $X_1, X_2, ..., X_N$ in order to guarantee a polynomial complexity with respect to the number of variables

[2] This definition differs from the one proposed in [67]. In fact, this definition does not require the CTBNC graph $\mathcal{G}$ to be connected. Therefore, feature selection is achieved as the product of any structural learning algorithm.

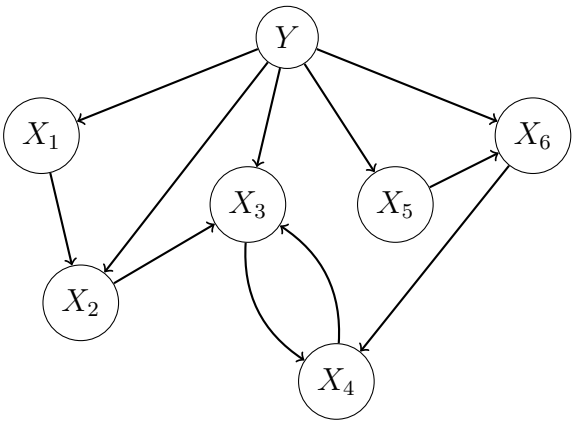


Fig. 2 Continuous time Bayesian network classifier; six attribute nodes $X_1, \ldots, X_6$ and the class node Y.

and the dimension of the data set [47]. In the case where k is not small a considerable computational effort is required to find the optimal graph structure $\mathcal{G}^*$. Therefore, we have to resort to hill-climbing optimization procedures to find an approximate solution to the considered optimization problem. Continuous time naive Bayes (CTNB) was introduced to limit the computational effort to find the optimal graph structure.

Definition 3 (Continuous time naive Bayes classifier). [67] A continuous time naive Bayes classifier is a continuous time Bayesian network classifier $\mathcal{C} = \{\aleph, P(Y)\}$ such that $Pa(X_n) = \{Y\}$, $n = 1, 2, ..., N$.

4.2 Classification

According to [67] a CTBNC $\mathcal{C} = \{\aleph, P(Y)\}$ classifies a *fully observed evidence stream* over J contiguous time intervals $\left(\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^J\right)$, i.e. a stream of continuous time evidence $\mathbf{X}^{[t_1,t_2)} = \mathbf{x}^{[t_1,t_2)} = \mathbf{x}^1$, $\mathbf{X}^{[t_2,t_3)} = \mathbf{x}^{[t_2,t_3)} = \mathbf{x}^2$, $\ldots$, $\mathbf{X}^{[t_J,t_{J+1})} = \mathbf{x}^{[t_J,t_{J+1})} = \mathbf{x}^J$ where all attribute nodes X_n, $n = 1, 2, ..., N$ are observed, by selecting the value y^* for the class Y which maximizes the posterior probability

$$P(Y|\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^J),$$

which is proportional to

$$P(Y) \prod_{j=1}^{J} q^{pa(X_{[j]})}_{x^j_{[j]} x^{j+1}_{[j]}} \prod_{n=1}^{N} exp\left(-q^{pa(X_n)}_{x^j_n} \delta_j\right), \tag{5}$$

where:

– $X_{[j]}$ is the variable transitioning at time t_{j+1};

- $q^{pa(X_{[j]})}_{x^j_{[j]}x^{j+1}_{[j]}}$ is the parameter associated with the transition from state $x^j_{[j]}$, in which the variable $X_{[j]}$ was during the j^{th} time interval, to state $x^{j+1}_{[j]}$, in which the variable $X_{[j]}$ will be at time t_{j+1}, given the state $pa(X_{[j]})$ of its parents during the j^{th} and the $(j+1)^{th}$ time interval;
- $q^{pa(X_n)}_{x^j_n}$ is the parameter associated with state x^j_n, in which the variable X_n was during the j^{th} time interval, given the state $pa(X_n)$ of its parents during the j^{th} time interval,

while $\delta_j = t_{j+1} - t_j$ is the length of the j^{th} time interval of the stream of continuous time evidence $\left(\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^J\right)$.

The learning algorithm based on marginal log-likelihood estimation for CTNB, and the inference algorithm for CTBNCs are described in [67].

4.3 Learning with log-likelihood and conditional log-likelihood

Structural learning for CTBNs is polynomial with respect to the number of variables and the size of the data set, once fixed the maximum number of parents (i.e. k). Nevertheless, increasing k, rapidly causes considerable computational efforts, while it implies more data is necessary to learn the node's parameters value conditioned on the possible parents instantiations. To overcome this limitation without making the restrictive assumption of conditional independence associated with the CTNB, the following instances from the class of CTBNCs have been proposed in [9,11].

Definition 4 (Max-k Continuous Time Bayesian Network Classifier [9,11]). A max-k continuous time Bayesian network classifier is a couple $\mathcal{M} = \{\mathcal{C}, k\}$, where $\mathcal{C}$ is a continuous time Bayesian network classifier $\mathcal{C} = \{\aleph, P(Y)\}$ such that the number of parents $|Pa(X_n)|$ for each attribute node X_n is bounded by a positive integer k. Formally, the following condition holds; $|Pa(X_n)| \leq k$, $n = 1, 2, ..., N$, $k > 0$.

Definition 5 (Max-k Augmented Continuous Time Naive Bayes [9,11]). A max-k augmented continuous time naive Bayes classifier is a max-k continuous time Bayesian network classifier such that the class node Y belongs to the parent set of each attribute node X_n, $n = 1, 2, ..., N$. Formally, the following condition holds; $Y \in Pa(X_n)$, $n = 1, 2, ..., N$.

Learning a continuous time Bayesian network classifier from data consists of learning a continuous time Bayesian network model where a specific node, i.e. the class node Y, does not depend on time. The constraint that the class node Y must have no parents simplifies the learning problem. Indeed, in such a particular case, the learning algorithm runs, for each attribute node X_n, $n = 1, 2, ..., N$, a local search to find its optimal set of parents, i.e. the set of parents which maximizes a given score function. Furthermore, for each attribute node X_n $n = 1, 2, ..., N$, no more than k parents are selected.

The structural learning algorithm proposed in [49] uses the Bayesian score function (4). The choice of the scoring function to be optimized is fundamental for structural learning. Nodelman et al [49] proposed the Bayesian scoring function which is based on the marginal log-likelihood estimation. The same learning algorithm can be adapted for learning a continuous time Bayesian network classifier.

Scoring functions based on log-likelihood are not the only suitable approaches to learning the structure of a CTBN classifier. Following what presented and discussed in [26], the log-likelihood function:

$$LL(\mathcal{M} \mid \mathcal{D}) = \sum_{i=1}^{|\mathcal{D}|} \log P_{\aleph}(y_i \mid \mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i}) + \log P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i}) \tag{6}$$

consists of two components; $\log P_{\aleph}(y_i \mid \mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i})$, which measures the classification *capability* of the model, and $\log P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i})$, which models the *dependencies between the attribute nodes* $X_1, X_2, ..., X_N$.

In [26] the authors remarked that in the case where the number of the attribute nodes X_n, $n = 1, 2, ..., N$ is large, the contribution to the scoring function value of $\log P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i})$ overwhelms the contribution of $\log P_{\aleph}(y_i \mid \mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i})$. However, the contribution of $\log P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i})$ is not directly related to the classification accuracy achieved by the classifier. Therefore, to improve the classification performance, Friedman et al [26] suggested to use the *conditional log-likelihood* as the scoring function. In such a case the maximization of the conditional log-likelihood aims to maximize the classification performance of the model without paying specific attention to the discovery of the existing dependencies between the attribute nodes X_n, $n = 1, 2, ..., N$.

In the case where models from the class of continuous time Bayesian network classifiers are considered, the conditional log-likelihood function can be written as follows [9]:

$$\begin{aligned}
CLL(\mathcal{M} \mid \mathcal{D}) &= \sum_{i=1}^{|\mathcal{D}|} \log P_{\aleph}(y_i \mid \mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i}) \\
&= \sum_{i=1}^{|\mathcal{D}|} \log \left(\frac{P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i} \mid y_i) P_{\aleph}(y_i)}{P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i})} \right) \\
&= \sum_{i=1}^{|\mathcal{D}|} \log \left(P_{\aleph}(y_i) \right) \\
&\quad + \sum_{i=1}^{|\mathcal{D}|} \log \left(P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i} \mid y_i) \right) \\
&\quad - \sum_{i=1}^{|\mathcal{D}|} \log \left(\sum_{y'} P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i} \mid y') P_{\aleph}(y') \right).
\end{aligned} \tag{7}$$

It is clear from (7) that the conditional log-likelihood function consists of the following three terms: the *class probability term* (8), the *posterior probability*

term (9), and finally the *denominator term* (10). These three terms can be estimated by using the available data set $\mathcal{D}$ as described in the following.

The *class probability term* is estimated as follows:

$$\sum_{i=1}^{|\mathcal{D}|} \log\left(P_{\aleph}(y_i)\right) = \sum_{y} M[y] \log(\theta_y) \tag{8}$$

where $M[y]$ is the number of trajectories of the data set $\mathcal{D}$ associated with the class y and θ_y is the parameter associated with the prior probability of class y.

From Equation (5) we can write the following:

$$\begin{aligned} P_{\aleph}(\mathbf{x}^1, ..., \mathbf{x}^{J_i} \mid y) &= \prod_{j=1}^{J} q_{x^j_{[j]} x^{j+1}_{[j]}}^{pa(X_{[j]})} \prod_{n=1}^{N} exp\left(-q_{x^j_n}^{pa(X_n)} \delta_j\right) \\ &= \prod_{j=1}^{J} q_{x^j_{[j]}}^{pa(X_{[j]})} \theta_{x^j_{[j]} x^{j+1}_{[j]}}^{pa(X_{[j]})} \prod_{n=1}^{N} exp\left(-q_{x^j_n}^{pa(X_n)} \delta_j\right). \end{aligned}$$

Therefore, the *posterior probability term* can be estimated as follows:

$$\begin{aligned} \sum_{i=1}^{|\mathcal{D}|} \log\left(P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i} \mid y_i)\right) = \sum_{n=1}^{N} \sum_{x_n, pa(X_n)} M[x_n \mid pa(X_n)] \log\left(q_{x_n}^{pa(X_n)}\right) \\ -q_{x_n}^{pa(X_n)} T[x_n \mid pa(X_n)] + \sum_{x'_n \neq x_n} M[x_n, x'_n \mid pa(X_n)] \log(\theta_{x_n x'_n}^{pa(X_n)}). \end{aligned} \tag{9}$$

The *denominator term* is similar to the first two components. Unfortunately, because of the sum, the *denominator term* cannot be further decomposed, while the sufficient statistics allow us to write:

$$\begin{aligned} &\sum_{i=1}^{|\mathcal{D}|} \log\left(\sum_{y'} P_{\aleph}(\mathbf{x}_i^1, ..., \mathbf{x}_i^{J_i} \mid y') P_{\aleph}(y')\right) = \\ &= \log\left(\sum_{y'} \theta_{y'} \prod_{n=1}^{N} \prod_{x_n, pa'(X_n)} (q_{x_n}^{pa'(X_n)})^{M[x_n | pa'(X_n)]}\right. \\ &\left. \exp(-q_{x_n}^{pa'(X_n)} T[x_n \mid pa'(X_n)]) \prod_{x'_n \neq x_n} (\theta_{x_n x'_n}^{pa'(X_n)})^{M[x_n, x'_n | pa'(X_n)]}\right) \end{aligned} \tag{10}$$

where $pa(X_n) = \{\pi_n \cup y\}$, $pa'(X_n) = \{\pi_n \cup y'\}$, while π_n represents the instantiation of the non-class parents of the attribute node X_n.

Unfortunately, no closed form solution exists to compute the optimal value of the model parameters, i.e. those parameters values which maximize the conditional log-likelihood (7). Therefore, an approach similar to the one introduced in [31] is followed: parameters values are obtained by using the

marginal log-likelihood estimation, described in Section 3.3 and introduced in [49], while structural learning is performed by maximizing the conditional log-likelihood function (7).

5 CTBNs for Clustering

Continuous time Bayesian network classifiers solve the problem of continuous time multivariate trajectory classification when labeled data are available. However, in real world applications very often labeled data are not available, thus we are faced with the *clustering problem*: the problem to discover the *groups of similar trajectories* which are hidden in the available data.

5.1 Parameter learning

Classification and clustering require different parameter learning algorithms. Given the structure of the continuous time Bayesian network model, the parameters can be learned from the sufficient statistics in the same way as it is done for continuous time Bayesian network classifiers. The difference consists in estimating the sufficient statistics associated with the hidden class. From Definition 2, we know that the class is a root node. Therefore, the computation of the sufficient statistics differs from what is done for the classification task only for the class variable and for those variables where the parent set contains the class variable. For parametric learning the well known Expectation Maximization (EM) algorithm can be used [40]. EM consists of two steps: *Expectation* and *Maximization* which are repeated iteratively until a stopping criteria is met. The expectation step computes the expected sufficient statistics which are used in the maximization step which computes the parameter values by maximizing the likelihood.

5.1.1 Expectation

In the case of clustering of observable data the expectation step needs to calculate the sufficient statistics as in a classical learning process, with the exception for the class node and for those nodes whose parent set contains the class node.

There are two types of sufficient statistics: the *occurrence counts* (i.e. M) and the *time counts* (i.e. T)[3]. Because only the class node (a static node) is unobserved, all the times spent by the variables in their states are known. Both time counts and occurrence counts of attributes with the class in their parent set must be weighted for the class probability. The class prior distribution can be estimated from the class expected occurrences.

[3] Time count sufficient statistics refers to the time spent in a particular state by a variable given the state of its parents.

Let us assume that the EM iterative algorithm starts with a random instantiation of the model parameters. Then, the contributions to the sufficient statistics for trajectory $i \in \{1, \ldots, \mid \mathcal{D} \mid\}$ of the available data set $\mathcal{D}$ are as follows. The contribution to the class count sufficient statistics for the class assignment $Y = y$ is:

$$\bar{M}[y] = \bar{M}[y] + P(y \mid \mathbf{x}_i^1, \ldots, \mathbf{x}_i^{J_i}).$$

The contribution to the occurrence count sufficient statistics of attribute X_n such that $Y \in Pa(X_n)$ and $Y = y$ is:

$$\begin{aligned}\bar{M}[x_n, x'_n \mid pa(X_n)] = \bar{M}[x_n, x'_n \mid pa(X_n)] \\ + M^i[x_n, x'_n \mid pa(X_n)/y] \cdot P(y \mid \mathbf{x}_i^1, \ldots, \mathbf{x}_i^{J_i}),\end{aligned}$$

where $M^i[x_n, x'_n \mid pa(X_n)/y]$ is the number of times that the attribute X_n transitions from state x_n to state x'_n in the i^{th} trajectory when its parents except the class node (i.e. $Pa(X_n)/Y$) are set to $pa(X_n)/y$.

The contribution to the time count sufficient statistics for an attribute X_n such that $Y \in Pa(X_n)$ and $Y = y$ is:

$$\begin{aligned}\bar{T}[x_n \mid pa(X_n)] = \bar{T}[x_n \mid pa(X_n)] \\ + T^i[x_n \mid pa(X_n)/y] \cdot P(y \mid \mathbf{x}_i^1, \ldots, \mathbf{x}_i^{J_i}),\end{aligned}$$

where $T^i[x_n \mid pa(X_n)/y]$ is the amount of time that the attribute X_n stays in state x_n when its parents except the class node (i.e. $Pa(X_n)/Y$) are set to $pa(X_n)/y$ for the i^{th} trajectory.

Expected sufficient statistics (i.e. $\bar{M}$ and $\bar{T}$) can be calculated by summing the contributions of the sufficient statistics, occurrences (i.e. M^i) and times (i.e. T^i) for each trajectory $i \in \{1, \ldots, \mid \mathcal{D} \mid\}$ of the available data set $\mathcal{D}$.

5.1.2 Maximization

The maximization step is the same as for fully observable trajectories. Using Bayesian estimation the parameters are calculated as follows:

- $q_x^{pa(X)} = \frac{\alpha_x^{pa(X)} + \bar{M}[x|pa(X)]}{\tau_x^{pa(X)} + \bar{T}[x|pa(X)]}$;
- $\theta_{xx'}^{pa(X)} = \frac{\alpha_{xx'}^{pa(X)} + \bar{M}[x,x'|pa(X)]}{\alpha_x^{pa(X)} + \bar{M}[x|pa(X)]}$;
- $\theta_y = \frac{\alpha_y + \bar{M}[y]}{\sum_{y'} \alpha_{y'} + \bar{M}[y']}$.

where $\bar{M}[x \mid pa(X)] = \sum_{x'} \bar{M}[x, x' \mid pa(X)]$.

5.2 Structural learning

Learning the structure of CTBNCs using unlabeled data can be done using the same algorithm as the one used for the classification task. Starting from an initial structure, an optimization algorithm can be used to locally maximize a given scoring function, as it is done in the case of classification.

However, learning the structure in case of clustering is a computationally demanding procedure that requires to iterate the EM parameter learning and the structural learning steps since a termination criteria is met. The optimization procedure is terminated when changing the model's structure does not improve the score with respect to the current model parameter values and sufficient statistics. It is worthwhile to mention that the structural learning process can be addressed independently for each variable.

6 Numerical experiments

Performance achieved by CTNB, Max-2 ACTNB, Max-2 CTBNC, Max-3 CTBNC, and Max-4 CTBNC, learned with the marginal log-likelihood and the conditional log-likelihood, are compared to performance achieved by DBNs for classification and clustering. The comparison is based on synthetic and real world data sets. Continuous time Bayesian network models are associated with a suffix specifying the scoring function which has been used for their learning. Suffix MLL is associated with the marginal log-likelihood scoring while suffix CLL is associated with the conditional log-likelihood scoring. To compare the continuous time models, two naive Bayes models are tested, where each variable depends on itself at the previous time step; while the first model (DBN-NB1) allows intra-slice relations (Figure 3a), the second model (DBN-NB2) allows relations between consecutive time slices (Figure 3b).

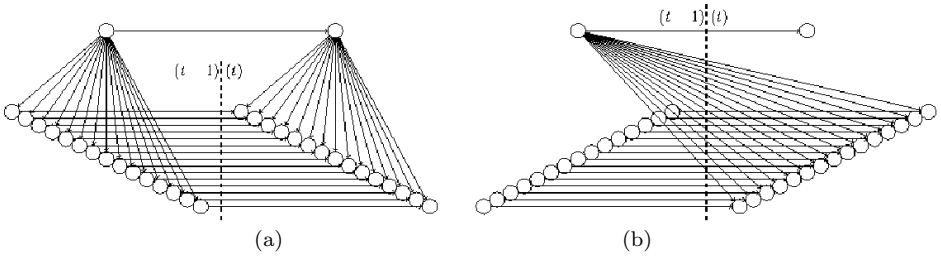


Fig. 3 DBN-NB1 (a) and DBN-NB2 (b) tested models.

Section 6.1 concerns continuous time Bayesian network classifiers. Their performance is analyzed and compared to that achieved by dynamic Bayesian network classifiers (DBNCs) on a real world data set concerning the post-stroke rehabilitation problem. Detailed analysis and comparison on synthetic data can be found in [9,11].

Section 6.2 concerns continuous time Bayesian network models for clustering. Their performance is analyzed and compared to that achieved by dynamic Bayesian network models on synthetic and real world data sets. Subsection 6.2.1 introduces the performance measures used for analysis and comparison purposes. Performance analysis and comparison on synthetic data sets are described in Subsection 6.2.2 where the data are sampled from CTBN classifiers and in Subsection 6.2.3 where the data are sampled from DBNs. Numerical experiments concerning the urban traffic profiling problem are described and analyzed in Subsection 6.2.4.

It is worthwhile to mention that in all the data sets (classification and clustering data sets) the classes/groups are balanced, i.e. classes/groups are equally probable. Numerical experiments on CTBNCs have been performed using the CTBNCToolkit [10], an open source Java toolkit developed by the authors to address and solve the problem of continuous time multivariate trajectory classification and clustering. Discrete time models have been implemented in MATLAB using the Bayesian Nets toolbox [46].

6.1 Classification

Tormene et al in [70] address the problem of post-stroke rehabilitation using the 1-Nearest Neighbor Classifier algorithm (NNC) applied to dynamic time warping (DTW) [39] and to open end DTW (OE-DTW) distances (see Section 2.2).

In this paper we focus on the post-stroke classification problems of 2 and 6 classes; while Tormene et al [70] performed the leave one out cross-validation, we decided to perform the 10-fold cross-validation. It is worthwhile to mention that in the Bayesian score (Equation (4)), the prior distribution over the structure (i.e. $\ln P(\mathcal{G})$) becomes less relevant when the amount of learning data increases. In the case where the number of learning samples belonging to the data set tends to infinity (i.e. $\mid \mathcal{D} \mid \rightarrow \infty$), the Bayesian score is equivalent to the marginal log-likelihood score (i.e. $\mathbf{MLLscore}\left(\mathcal{G} : \mathcal{D}\right) = \ln P(\mathcal{D}|\mathcal{G})$). To fairly compare the classification performance achieved when using the conditional log-likelihood score (Equation (7)), which does not use any graph structure penalization term, the marginal log-likelihood score is used instead of the Bayesian score.

The parameters of the models are set using an empirical procedure. The amount of data in the post-stroke rehabilitation data sets is limited. When the 6 class problem is analyzed, this shortage of data becomes evident from the analysis of the model parameters. The selected values for the imaginary counts have a greater impact on the sufficient statistic. We noticed that the performances achieved by the CTBNCs learned with the CLL score are robust with respect to the value selected for the imaginary counts, while the same does not happen for the MLL scoring. $\alpha_{xx'} = 1$, $\alpha_y = 1$ ([4]) and $\tau_x = .005$ are the

[4] With α_y we refer to the hyperparameter associated with the class value y.

parameters set for the CLL scoring. We observed that changing these parameters values does not impact too much the performance. Different parameters values have been used for the MLL scoring to achieve acceptable performance values for the 6 classes problem. In this case small variations of the parameters values greatly affect the model performances.

Table 1 shows the accuracy values, averaged over the movements, achieved by CTBNCs and DBNCs for 2 and 6 classes classification problems. In bold showed the performances achieved by the classifiers belonging to the set of best classifiers calculated in accord with the following procedure. Let $A_1, A_2, ..., A_r$ be the (sample) average accuracy values achieved on a given data set $\mathcal{D}$ by the classifiers $\mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_r$. The *set of best classifiers* $\mathbb{C}^\alpha$ at the α significance level is obtained according to the following procedure: **1)** Sort $A_1, A_2, ..., A_r$ in descending order to obtain $A_{(1)}, A_{(2)}, ..., A_{(r)}$, **2)** Set the current number of tested hypothesis to one, i.e. set $\tau := 1$, and initialize the *set of best classifiers* $\mathbb{C}^\alpha$ with the classifier $\mathcal{C}_{(1)}$ achieving the best average accuracy value $A_{(1)}$, i.e. set $\mathbb{C}^\alpha := \mathcal{C}_{(1)}$, **3)** If $\tau = r$ then the procedure terminates. Otherwise, perform a *paired t-test* at the α/τ (Bonferroni's correction) significance level, where the null hypothesis compares the true and unknown accuracy $\mu_{(1)}$ associated with the classifier $\mathcal{C}_{(1)}$ to the true and unknown accuracy $\mu_{(\tau+1)}$ associated with the classifier $\mathcal{C}_{(\tau+1)}$. In particular, the null hypotheis is as follows $H_0 : \mu_{(1)} = \mu_{(\tau+1)}$ while the alternative hypothesis is as follows $H_1 : \mu_{(1)} > \mu_{(\tau+1)}$, **4)** If H_0 is not rejected then update the *set of best classifiers* as follows $\mathbb{C}^\alpha := \mathbb{C}^\alpha \cup \mathcal{C}_{(\tau+1)}$, increase the number of the tested hypothesis as follows $\tau := \tau + 1$ and go to step **3)**. Otherwise, if the null hypothesis H_0 is rejected then the procedure terminates.

# classes	CTNB	Max-2 ACTNB (MLL)	Max-2 ACTNB (CLL)	Max-2 CTBNC (MLL)	Max-2 CTBNC (CLL)	Max-3 CTBNC (MLL)	Max-3 CTBNC (CLL)	Max-4 CTBNC (MLL)	Max-4 CTBNC (CLL)	DBN-NB1	DBN-NB2
2 classes	**0.98**	0.97	**0.99**	0.87	0.85	0.87	0.92	0.87	0.95	0.97	0.97
6 classes	**0.91**	**0.91**	**0.89**	0.81	**0.88**	0.81	**0.88**	0.81	**0.88**	0.87	0.87

Table 1 Average accuracy for the post-stroke rehabilitation data sets (10-fold CV). Bold digits are associated with classifiers belonging to the sets of best classifiers $\mathbb{C}^{0.05}$ at the 5% significance level.

CTBNCs performances are comparable to those achieved by DTW [70], even after discretization of the original state space. Accuracy values achieved by almost all the CLL classifiers are significantly better than those achieved by theirs MLL counterparts at the 5% significance level. For the 6 class classification problem, in the case where no information about variable dependency is available (i.e. the links between the class and the other variables), CLL always outperforms MLL at the 5% significance level.

Behavior of CTBNCs was investigated with respect to the amount of available data. Reduced data sets were obtained by limiting both the length and the number of trajectories, which is different from what has been done in [70], where only the length of the trajectories of the test set was cut, while the trajectories of the training set were full length. While Tormene et al [70] were

interested in analyzing the behavior of DTW and OE-DTW during movement execution we are interested in analyzing the behavior of the CTBNCs' learning process with respect to the amount of available data (Figure 4). The trajectories do not contain missing data.

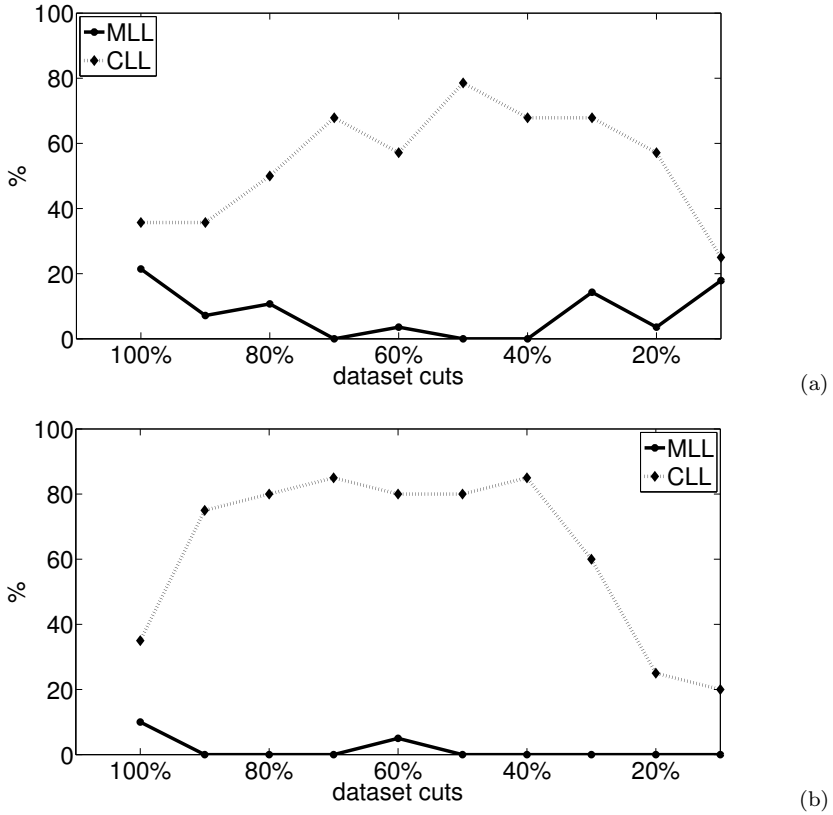


Fig. 4 Percentage of numerical experiments where the MLL (CLL) achieved an accuracy value which is greater than the accuracy value achieved by the CLL (MLL) at the 5% significance level. Analysis is performed on post-stroke and with respect to reduced data sets. The x-axes represent data sets percentage reduction. Figure **(a)** refers to the 2 class problem while Figure **(b)** refers to the 6 class problem.

Analyzing Figure 4(a), which is associated with the 2 class classification problem, and Figure 4(b), which is associated with the 6 class classification problem, we conclude that CLL outperforms MLL. This confirms what emerged in [9,11] from numerical experiments on synthetic data sets. Effectiveness of CLL becomes more evident in the case where the amount of available data is reduced. In this case CLL exploits its effectiveness to detect weak dependencies between feature nodes and the class variable which allow CTBNCs learned

with CLL to achieve the best accuracy. However, in the case where available data are too few, the CTNB classifier is the best option[5].

6.2 Clustering

6.2.1 Performance measures

Performance evaluation and comparison of clustering algorithms/models can be done by using *external measures*, *internal measures* or, *relative measures* [35, 28,77]. Internal and external measures are based on statistical tests. Relative measures are not based on statistical tests, and for this reason they are more efficient [35]. Internal measures evaluate the similarity of the clusters using distance measures. The advantage of internal measures is the possibility to obtain performance measures also when the label is unknown in the evaluation data set. On the contrary, if it is difficult to calculate a meaningful distance measure, internal measures lose their effectiveness. This is the case of continuous time multivariate trajectories, where it is not clear what is a meaningful distance measure. External measures evaluate clustering models/algorithms by exploiting the available label information. While the label is ignored during the learning process, it is used to compute the performance achieved by the learnt model/algorithm. Because the labels are available for synthetic and real world data sets, the following external measures have been used: *Rand index* (R) [57], *Jaccard's coefficient* (J) [35] and, *Fowlkes-Mallows index* (FM) [25].

These measures can be calculated as follows. Consider the following clustering partition $\mathbf{C} = \{C_1, \ldots, C_h\}$ and let $\mathbf{P} = \{P_1, \ldots, P_s\}$ be the true partition. Given a couple of clustered trajectories of the data set it is possible to distinguish the following four cases [36]:

- SS: trajectories belong to the same cluster and to the same partition;
- SD: trajectories belong to the same cluster, but to different partitions;
- DS: trajectories belong to different clusters, but to the same partition;
- DD: trajectories belong to different clusters and to different partitions.

Consider a data set $\mathcal{D}$ consisting of $\mid \mathcal{D} \mid$ instances and all the possible couples of instances M in the data set (i.e. $M = \frac{|\mathcal{D}|(|\mathcal{D}|-1)}{2}$), and let $\#SS$, $\#SD$, $\#DS$ and, $\#DD$ be respectively the number of couples in each of the four possible configurations (i.e. $M = \#SS + \#SD + \#DS + \#DD$). Then, the number of occurrences in each of the four cases provides useful information for evaluating the quality of the clustering partition $\mathbf{C}$. Starting from these values the following external measures are calculated [36]:

- Rand index (R): $R = \frac{\#SS+\#DD}{M}$
- Jaccard's coefficient (J): $J = \frac{\#SS}{\#SS+\#SD+\#DS}$
- Fowlkes-Mallows index (FM): $FM = \sqrt{\frac{\#SS}{\#SS+\#SD}\frac{\#SS}{\#SS+\#DS}}$

[5] For further experiments on continuous time Bayesian network classifiers for classification purposes refer to [9,11].

Figure	# classes white	qs range light gray	qs range gray	qs range dark gray	Figure	# classes white	qs range light gray	qs range gray	qs range dark gray
CTNB					Max-2 ACTNB				
5a	4	$[1,2]$	$[2,4]$	$[4,8]$	5b	4	$[1,2]$	$[2,4]$	$[4,8]$
5a	4	$[1,2]$	$[4,8]$	$[8,16]$	5b	4	$[10,20]$	$[20,40]$	$[40,80]$
5a	4	$[10,20]$	$[20,40]$	$[40,80]$	5b	10	$[10,20]$	$[20,40]$	$[40,80]$
5a	10	$[1,2]$	$[2,4]$	$[4,8]$	5c	4	$[1,2]$	$[2,4]$	$[4,8]$
5a	10	$[10,20]$	$[20,40]$	$[40,80]$	5c	4	$[10,20]$	$[20,40]$	$[40,80]$
Max-2 CTBNC					Max-3 CTBNC				
5d	4	$[1,2]$	$[2,4]$	$[4,8]$	5g	4	$[1,2]$	$[2,4]$	$[4,8]$
5d	4	$[10,20]$	$[20,40]$	$[40,80]$	5g	4	$[10,20]$	$[20,40]$	$[40,80]$
5e	4	$[1,2]$	$[2,4]$	$[4,8]$	5h	4	$[1,2]$	$[2,4]$	$[4,8]$
5e	4	$[10,20]$	$[20,40]$	$[40,80]$	5h	4	$[10,20]$	$[20,40]$	$[40,80]$
5f	4	$[1,2]$	$[2,4]$	$[4,8]$	5i	4	$[1,2]$	$[2,4]$	$[4,8]$
5f	4	$[10,20]$	$[20,40]$	$[40,80]$	5i	4	$[10,20]$	$[20,40]$	$[40,80]$
Max-4 CTBNC									
5j	4	$[1,2]$	$[2,4]$	$[4,8]$					
5j	4	$[10,20]$	$[20,40]$	$[40,80]$					
5k	4	$[1,2]$	$[2,4]$	$[4,8]$					
5k	4	$[10,20]$	$[20,40]$	$[40,80]$					
5l	4	$[1,2]$	$[2,4]$	$[4,8]$					
5l	4	$[10,20]$	$[20,40]$	$[40,80]$					

Table 2 Summary of the CTBNC tested structures. *Figure* specifies the figure number of the structure. *# classes* represents the number of classes. The remaining columns show the sampling interval for the value of the parameter q for each node, depending on its gray tonality as depicted in Figure 5.

These measures take value in $[0,1]$. The more similar the clustering partition $\mathbf{C}$ is to the true partition $\mathbf{P}$, the higher the values of the three measures are.

6.2.2 Continuous time synthetic data

The value of the rand index, Jaccard's coefficient and Fowlkes-Mallows index achieved by different instances of CTBNCs and DBNCs on synthetic data sets are compared. Synthetic data, are generated by sampling from CTBNC models of increasing complexity. To apply DBNCs to continuous time trajectories, the time evolution of the trajectories has been discretized by using a sampling rate which generates a time slice for each sample. Data sets consist of $1,000$ trajectories with average length ranging from 300 (CTNBs) to 1,400 (Max-4 CTBNCs). Analyzed models' structures are CTNB (Figure 5(a)), Max-2 ACTNB (Figure 5(b,c)), Max-2 CTBNC (Figure 5(d-f)), Max-3 CTBNC (Figure 5(g-i)), and Max-4 CTBNC (Figure 5(j-l)).

For each structure, different assignments of parameters value are sampled from a given interval. Each pair, (*model's structure*, *parameters value*), is used to generate a learning data set. Table 2 summarizes the tested synthetic data sets. The table shows the structure, the class cardinality and the sampling intervals for each test. For each model category 10 different parameters samples were generated.

CTBNCs are developed in Java [10], while DBNC experiments use naive Bayes models implemented by the MATLAB Bayesian Nets toolbox [46]. DBN-NB1 and DBN-NB2 (Figure 3) are the discrete time models used for comparison.

Model parameters values have been selected by exploiting a small amount of the synthetic data. A complete investigation on each model was not feasible

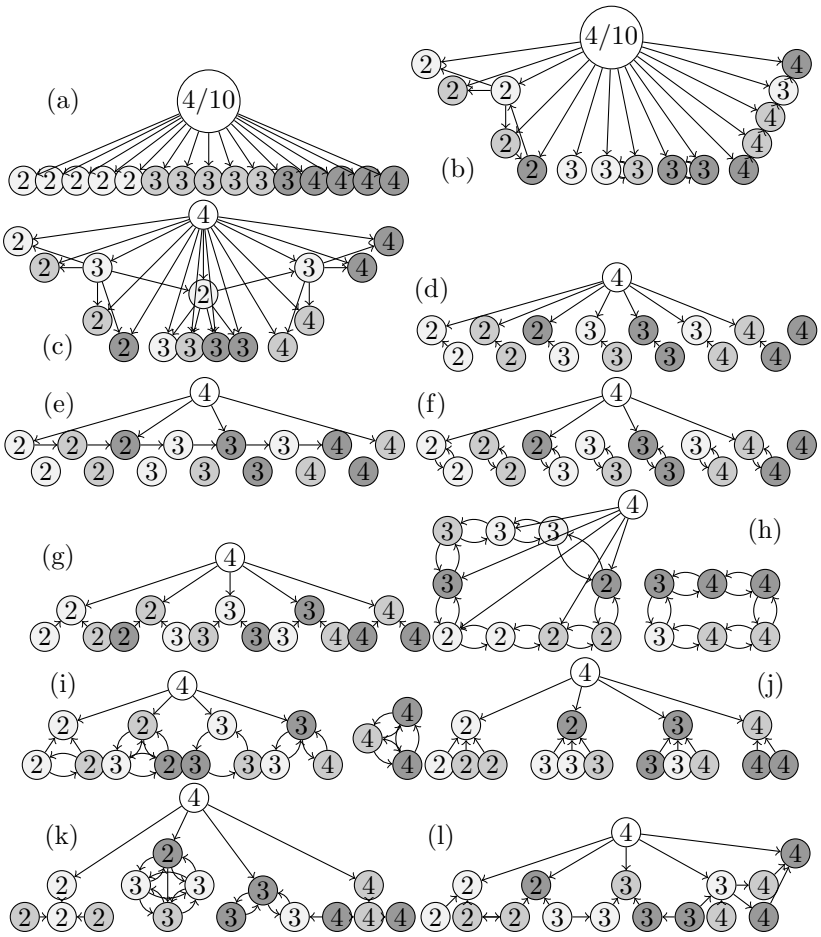


Fig. 5 CTNB (a), Max-2 ACTNB (b,c), Max-2 CTBNC (d-f), Max-3 CTBNC (g-i) and, Max-4 CTBNC (j-l) tested structures. Numbers associated with nodes represent the cardinality of the corresponding variables. If more than one value is reported, the corresponding cardinality values have been tested. White nodes are associated with classes, while the darker the gray color is, the wider is the interval where the q parameter values are sampled (see Table 2).

due to time requirements. Regarding CTBNCs the best parameters values for synthetic data sets are $\alpha_{xx'} = 1$, $\alpha_y = 1$ ([6]) and $\tau_x = .005$. These parameters values perform well for both marginal and conditional log-likelihood scores.

The average performance are summarized in Table 3 and Figure 6. The rows of Table 3 are associated with the type of the model from which the data set has been sampled. A data set obtained by sampling from a Max-v CTBNC model will be referred to as KvCTBNC.

Figure 6 depicts the best average values of Rand index (R), Jaccard's coefficient (J), and Fowlkes-Mallows index (FM). For each measure and for

[6] With α_y we refer to the hyperparameter associated with the class value y.

Test	Measure	CTNB	$k=2$ ACTNB (MLL)	$k=2$ ACTNB (CLL)	$k=2$ CTBNC (MLL)	$k=2$ CTBNC (CLL)	$k=3$ CTBNC (MLL)	$k=3$ CTBNC (CLL)	$k=4$ CTBNC (MLL)	$k=4$ CTBNC (CLL)	DBN-NB1	DBN-NB2
CTNB	R	**0.96**	**0.96**	0.89	0.76	0.92	0.75	0.79	0.77	0.70	0.59	0.41
	J	0.77	**0.80**	0.51	0.54	0.60	0.51	0.25	0.58	0.11	0.15	0.17
	FM	0.86	**0.88**	0.61	0.67	0.69	0.63	0.37	0.72	0.19	0.27	0.34
K2ACTNB	R	0.82	**0.87**	0.80	0.67	0.81	0.69	0.67	0.69	0.66	0.55	0.32
	J	0.39	**0.55**	0.37	0.41	0.39	0.44	0.13	0.46	0.13	0.16	0.20
	FM	0.54	**0.65**	0.49	0.57	0.51	0.60	0.23	0.61	0.22	0.30	0.41
K2CTBNC	R	0.69	0.79	0.68	0.80	0.66	**0.84**	0.63	**0.84**	0.63	0.56	0.44
	J	0.25	0.42	0.23	0.45	0.20	**0.52**	0.15	**0.52**	0.14	0.17	0.21
	FM	0.39	0.59	0.37	0.60	0.33	**0.68**	0.26	**0.68**	0.25	0.30	0.38
K3CTBNC	R	0.65	0.69	0.64	0.68	0.63	0.86	0.63	**0.88**	0.63	0.58	0.37
	J	0.17	0.24	0.16	0.35	0.16	0.58	0.15	**0.63**	0.14	0.17	0.23
	FM	0.29	0.38	0.28	0.53	0.27	0.72	0.26	**0.77**	0.25	0.29	0.43
K4CTBNC	R	0.72	0.79	0.69	0.67	0.70	0.71	0.65	**0.80**	0.63	0.58	0.44
	J	0.32	0.46	0.25	0.38	0.27	0.63	0.17	**0.65**	0.14	0.17	0.21
	FM	0.44	0.58	0.38	0.52	0.40	0.75	0.29	**0.76**	0.25	0.29	0.39

Table 3 Average clustering performances for each class of model used to generate the data sets. R stands for the Rand index, J stands for Jaccard's coefficient and FM for the Fowlkes-Mallows index. In bold the greatest value for each row.

each class of models (i.e. MLL, CLL, CTNB, and DBN) the best average performances are reported.

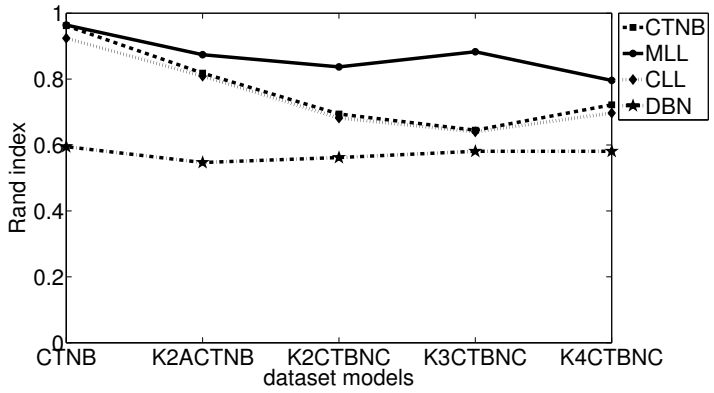
Figure 6 clearly shows the effectiveness of continuous time Bayesian network classifiers learned by maximizing the marginal log-likelihood scoring function. It is worthwhile to mention that for the clustering task the conditional log-likelihood score does not perform well as it happens for the classification task [9]. Indeed, CTBNCs learned with conditional log-likelihood perform comparably or worse than CTNB.

Dynamic Bayesian networks are significantly outperformed by the continuous time models. DBN performances are clearly worse than those achieved by continuous time models which are learned by maximizing the marginal log-likelihood score. This holds true also with respect to the continuous time naive Bayes classifier.

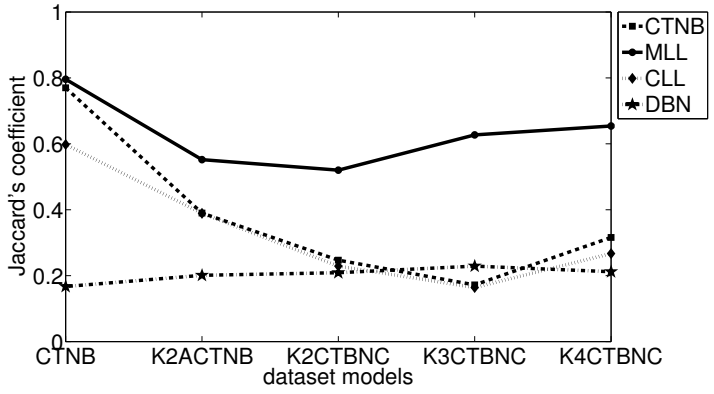
Figure 7 depicts the average learning time for the continuous time Bayesian network models. Because the models were compared on continuous time data sets, to apply discrete time models it was necessary to discretize the trajectory timing. For each trajectory a sampling rate has been chosen in order to generate 50 time slices. The discretization process makes the learning time of CTBNCs and DBNs not directly comparable. Nevertheless, learning DBNs requires a greater computational time than learning a CTBNCs.

Among continuous time models, CTNB is of course the most efficient one because it does not require to perform the structural learning. The main differences in terms of learning time between CTBNCs emerge when the maximum number of parents is increased. Continuous time Bayesian networks learned by maximizing marginal log-likelihood seem to be learned faster than the corresponding models learned by maximizing the conditional log-likelihood.

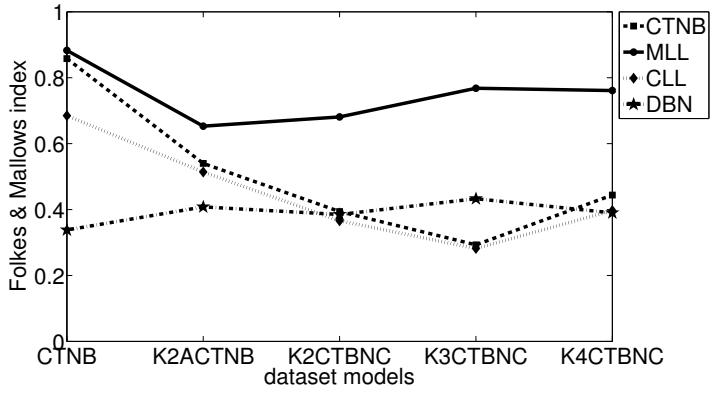
The best compromise between complexity and effectiveness is offered by $k = 2$ ACTNB which despite the limited number of parents is able to provide



(a)



(b)



(c)

Fig. 6 Best average performances, i.e. Rand index (a), Jaccard's coefficient (b), and Fowlkes-Mallows index (c), between MLL, CLL, CTNB, and DBN models. The x-axis is associated with the CTBNCs from which the data sets have been sampled.

good performances even when the complexity of the problem increases (see Table 3).

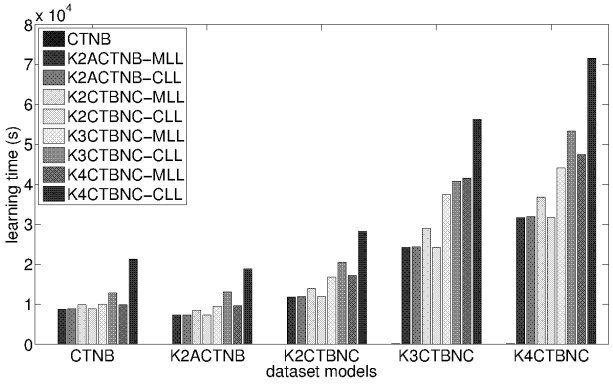


Fig. 7 Average learning time for each CTBNC model. The x-axis is associated with the models used to generate the data sets.

6.2.3 Discrete time synthetic data

To fairly compare CTBNCs with DBNCs, synthetic data sets are sampled from five different DBN models. While in the previous tests DBNCs required a time discretization, in this case we are comparing the approaches on discrete time trajectories where DBNCs are naturally applicable.

Figure 8 depicts two of the DBN models used to generate the discrete time data sets. The other models are variants of them, where the maximum number of parents ranges from 2 to 4. Parameters associated with the DBNC's nodes are sampled from the Dirichelet distribution, except for the first time slice and for the class node whose parameters values are sampled from the uniform distribution. Since the CTBNCs make inference about a class that will occur in the future, DBNC data sets have been sampled from models where the class value does not change over time.

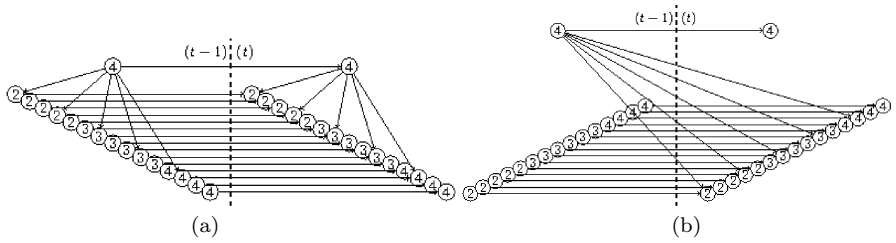


Fig. 8 Example of DBN models used to generate discrete time data sets. White nodes are associated with classes. Numbers represent the cardinality of each node. Figure (a) depicts the structure of DBNTest1 model. DBNTest2 has the same class relations, but feature nodes in the first time slice are connected with 2 nodes of the next time slice. Figure (b) depicts the structure of DBNTest3. DBNTest4 and DBNTest5 have the same class relations, but features nodes in the first time slice are connected with 2 (DBNTest4) and 3 (DBNTest5) nodes of the next time slice.

All five data sets consist of 100 trajectories of 20 time slices each. As in the case of continuous time trajectories, the parameters of the tested models are set after an empirical analysis. The hyperparameters used in the case of discrete time trajectories correspond to the ones used in the case of continuous time trajectories.

Results achieved on discrete time synthetic data sets agree with what observed on continuous time synthetic data sets. Figure 9 depicts the best values of Rand index (R), Jaccard's coefficient (J), and Fowlkes-Mallows index (FM). For each measure and for each class of models (i.e. MLL, CLL, CTNB, and DBN) the best performances are reported.

The synthetic data sets generated by dynamic Bayesian networks are composed of discrete time trajectories. Nevertheless, discrete time models (i.e. DBNs) do not perform well as it happens for continuous time models. As in the case of continuous time synthetic data sets, CTBNCs learned with the marginal log-likelihood score outperform all the other models. Continuous time Bayesian network models learned by maximizing conditional log-likelihood perform better than dynamic Bayesian network models, but their performances are inferior to those achieved by the continuous time naive Bayes classifier.

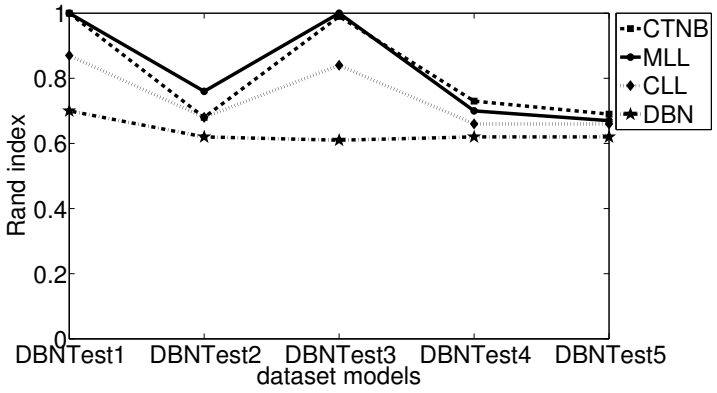
Test	Measure	CTNB	$k=2$ ACTNB (MLL)	$k=2$ ACTNB (CLL)	$k=2$ CTBNC (MLL)	$k=2$ CTBNC (CLL)	$k=3$ CTBNC (MLL)	$k=3$ CTBNC (CLL)	$k=4$ CTBNC (MLL)	$k=4$ CTBNC (CLL)	DBN-NB1	DBN-NB2
DBNTest1	R	**1**	**1**	0.87	0.26	0.62	0.92	0.59	0.90	0.61	0.70	0.55
	J	**1**	**1**	0.62	0.26	0.15	0.77	0.16	0.71	0.15	0.27	0.18
	FM	**1**	**1**	0.77	0.51	0.26	0.88	0.28	0.83	0.26	0.43	0.32
DBNTest2	R	0.68	**0.76**	0.67	0.25	0.62	0.25	0.62	0.25	0.63	0.62	0.50
	J	0.26	**0.36**	0.22	0.25	0.15	0.25	0.14	0.25	0.17	0.15	0.21
	FM	0.41	**0.52**	0.35	0.50	0.25	0.50	0.25	0.50	0.29	0.26	0.36
DBNTest3	R	0.99	**1**	0.84	0.91	0.25	0.86	0.60	0.85	0.25	0.61	0.44
	J	0.96	**1**	0.52	0.69	0.25	0.59	0.15	0.56	0.25	0.15	0.22
	FM	0.98	**1**	0.68	0.82	0.50	0.74	0.26	0.72	0.50	0.26	0.40
DBNTest4	R	**0.73**	0.69	0.66	0.27	0.63	0.27	0.62	0.27	0.64	0.62	0.55
	J	**0.31**	0.28	0.21	0.27	0.16	0.27	0.16	0.27	0.17	0.15	0.18
	FM	**0.47**	0.43	0.34	0.52	0.28	0.52	0.28	0.52	0.30	0.26	0.32
DBNTest5	R	**0.69**	0.67	0.66	0.26	0.63	0.26	0.63	0.26	0.62	0.62	0.32
	J	**0.24**	0.22	0.21	0.26	0.15	0.26	0.15	0.26	0.14	0.15	0.24
	FM	**0.38**	0.36	0.35	0.51	0.26	0.51	0.26	0.51	0.24	0.26	0.46

Table 4 Clustering performances for each DBN test and for each model. R stands for the Rand index, J stands for Jaccard's coefficient and FM for the Fowlkes-Mallows index. In bold the greatest value for each row.

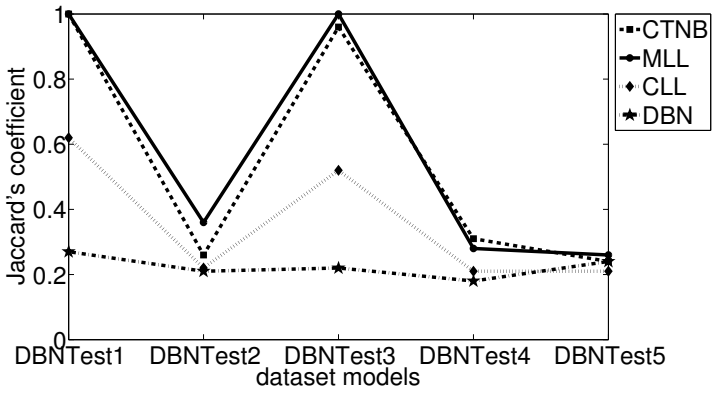
Table 4 summarizes the results obtained using the discrete time data sets. Also in this case, $k = 2$ ACTNB shows its capability to achieve good performances. In the case of clustering discrete time data sets, $k = 2$ ACTNB learned by maximizing the marginal log-likelihood is the model which achieves the best performances in many cases and, when it is not the best model, its performances are competitive with other models.

6.2.4 Clustering of urban traffic profiles

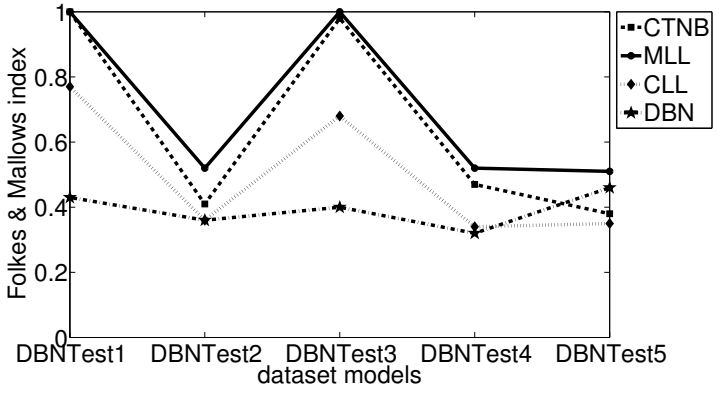
Clustering urban traffic data allows to discover traffic profiles, i.e. traffic trajectories over time. Traffic profiles are used to formulate improved instances,



(a)



(b)



(c)

Fig. 9 Best performances, i.e. Rand index (a), Jaccard's coefficient (b), and Fowlkes-Mallows index (c), between MLL, CLL, CTNB, and DBN models. The x-axis is associated with the DBN model from which the data sets have been sampled.

i.e. more efficient, of the traffic light cycles optimization problem (see Section 2.3). In urban traffic control it is customary to use loops which summarize the

measured square wave by the following attributes: vehicle counts, vehicle average speed and density in a given time interval. Such attributes are aggregated over fixed time intervals (e.g., five minutes). In this context we propose to use the square wave, a variable which measure in continuous time the presence of a vehicle over a loop, as an effective and efficient approach for urban traffic control. This is made possible by CTBNCs which allow to efficiently manage continuous time data.

Four data sets were generated using the TSIS-CORSIM simulator [14,51]. The data sets were generated using 100 seconds and 300 seconds as length of trajectories. Data were sampled at a frequency of 10Hz. Sensors (i.e. loops) were positioned at the beginning and at the end of each link (i.e. a road that links two crossroads). Two data sets from a toy network example and two data sets from a portion of the Monza's road network were generated (see Figure 10). The choice of this portion of the Monza's road network is particularly valuable because it is one of the most crucial and most congested areas in Monza and thus in northern Italy. This is confirmed by the partnership of Monza in the CIVITAS ARCHIMEDES European project[7] with the purpose of introducing "innovative, integrated and ambitious strategies for clean, energy-efficient, sustainable urban transport and thereby have a significant impact on policies concerning energy, transport, and environmental sustainability".

Performance of continuous time Bayesian network models in traffic profile clustering are analyzed. Since the real Monza road network is not equipped with two sensors for each link, tests are also made by using the actual six sensors installed on the road network.

Figure 11 depicts the best values of the Rand index (R), Jaccard's coefficient (J), and Fowlkes-Mallows index (FM). For each measure and for each class of models (i.e. MLL, CLL, CTNB, DBN) the best performances are shown[8].

The results, summarized in Table 5, show the effectiveness of continuous time classifiers. CTBNCs strongly outperform DBNs.

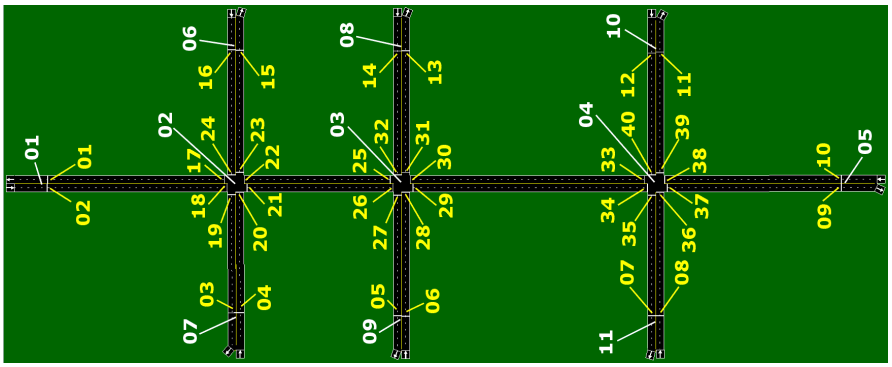
Continuous time Bayesian network models learned by maximizing the marginal log-likelihood are confirmed to be the best option to deal with the clustering problem. Nevertheless, continuous time naive Bayes and continuous time Bayesian network models learned by maximizing the conditional log-likelihood perform well. Dynamic Bayesian networks are outperformed by all the continuous time models. CTBNCs are more effective when using the continuous time sensor information and furthermore they have a lower computational complexity.

7 Conclusions

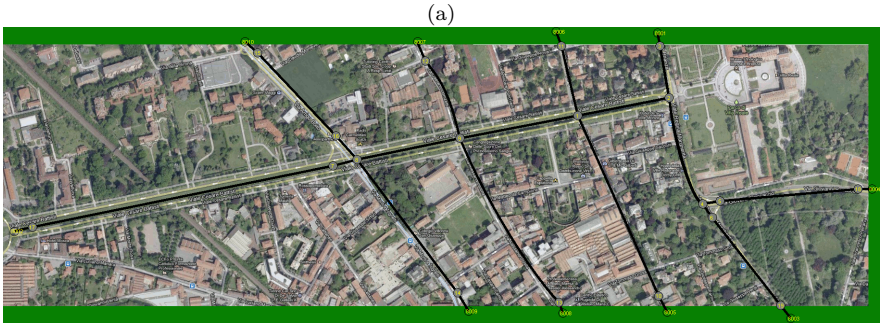
Continuous time Bayesian network models for classification and clustering of multivariate discrete state and continuous time streaming data are presented.

[7] http://www.civitas.eu/archimedes

[8] For computational reasons a random subset of trajectories is used in the Monza road network tests.



(a)



(b)

Fig. 10 Toy road network (a) and Monza's road network (b) used to generate the two data sets.

Road network	Trajectories length	Measure	CTNB	$k=2$ ACTNB (MLL)	$k=2$ ACTNB (CLL)	$k=2$ CTBNC (MLL)	$k=2$ CTBNC (CLL)	$k=3$ CTBNC (MLL)	$k=3$ CTBNC (CLL)	$k=4$ CTBNC (MLL)	$k=4$ CTBNC (CLL)	DBN-NB1	DBN-NB2
Toy netwok	100 seconds	R	**0.93**	**0.93**	**0.93**	**0.93**	0.87	0.92	**0.93**	0.86	**0.93**	0.48	0.64
		J	**0.70**	0.69	**0.70**	**0.70**	0.50	0.67	**0.70**	0.49	**0.70**	0.19	0.18
		FM	0.82	0.82	**0.83**	0.82	0.67	0.80	**0.83**	0.66	0.82	0.37	0.32
	300 seconds	R	0.82	0.87	0.88	0.90	0.95	**0.97**	0.91	0.96	0.88	0.49	0.31
		J	0.40	0.53	0.53	0.57	0.77	**0.85**	0.66	0.80	0.54	0.16	0.18
		FM	0.57	0.70	0.70	0.72	0.87	**0.92**	0.80	0.89	0.70	0.32	0.39
Monza's netwok*	100 seconds	R	0.80	**0.81**	0.80	0.80	0.80	0.79	0.80	0.79	0.80	0.25	0.34
		J	0.29	0.30	0.29	0.30	0.30	0.28	0.28	0.27	**0.31**	0.18	0.17
		FM	0.45	**0.47**	0.45	0.46	0.46	0.44	0.44	0.43	**0.47**	0.40	0.37
	300 seconds	R	0.82	0.82	0.80	0.81	**0.83**	0.82	0.81	0.82	0.82	0.53	0.54
		J	0.34	0.36	**0.38**	0.33	0.36	0.33	0.32	0.32	0.34	0.24	0.25
		FM	0.51	0.53	**0.55**	0.49	0.53	0.50	0.49	0.49	0.50	0.46	0.47
Monza's network real loops*	100 seconds	R	0.77	**0.78**	0.76	0.77	0.76	0.77	0.77	**0.78**	0.76	0.59	0.50
		J	0.21	**0.24**	0.21	**0.24**	0.22	0.22	0.23	**0.24**	0.22	0.14	0.16
		FM	0.35	**0.39**	0.34	**0.39**	0.37	0.36	0.37	0.38	0.36	0.27	0.30
	300 seconds	R	0.79	**0.81**	0.77	**0.81**	**0.81**	0.80	0.80	0.80	**0.81**	0.61	0.57
		J	0.26	0.30	0.24	0.30	0.29	0.27	0.29	**0.31**	**0.31**	0.19	0.20
		FM	0.41	0.46	0.39	0.47	0.45	0.43	0.46	0.47	**0.48**	0.35	0.37

Table 5 Clustering performances on the traffic profiling data sets. R stands for the Rand index (R), J stands for Jaccard's coefficient (J) and FM for the Fowlkes-Mallows index (FM). The asterisk indicates that for computational reasons a random subset of trajectories was used for the experiments. In bold the greatest value for each row.

Results achieved on synthetic data confirm effectiveness and efficiency of continuous time Bayesian network models when compared to dynamic Bayesian

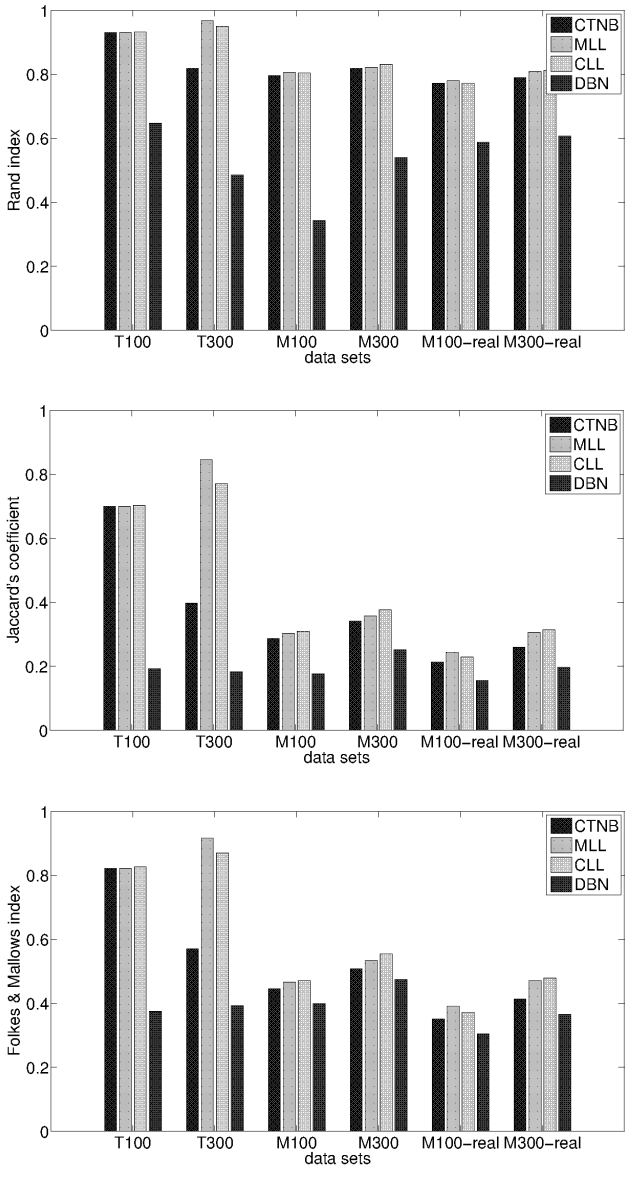


Fig. 11 The Rand index (a), Jaccard's coefficient (b), and Fowlkes-Mallows index (c), between MLL, CLL, CTNB, and DBN models. The x-axis is associated with the used data sets. "T100" and "T300" indicate the toy network data sets. "M100", "M300", "M100-real", and "M300-real" indicate the Monza road network data sets with all the sensors and with only the real sensors.

networks. Performances achieved by continuous time Bayesian network classifiers for the problem of post-stroke rehabilitation confirm their effectiveness.

Furthermore, continuous time Bayesian network models compared favourably to their discrete time counterpart, i.e. dynamic Bayesian networks, to solve the complex problem of learning urban traffic profiles from continuous time loop data. Results show that the conditional log-likelihood score is more effective than the marginal log-likelihood score when continuous time Bayesian networks are learnt to solve classification problems, while the marginal log-likelihood is more effective than the conditional log-likelihood score to solve clustering problems.

Numerical experiments on continuous time Bayesian networks have been performed using the CTBNCToolkit [10], an open source Java toolkit developed by the authors to address and solve the problem of continuous time multivariate trajectory classification and clustering.

8 Future works

Continuous time Bayesian network classifiers were recently introduced in the literature, therefore there are still open issues that can be taken into account. The memoryless property of the exponential distribution is a limitation for many real world applications. Therefore, a future direction consists in modelling state duration by using different distributions, i.e. phase-type distributions. Some work has been done in this direction for continuous time Bayesian networks [30, 50]. Interesting is also the application of the continuous time Bayesian network inference algorithm on Piecewise-constant Conditional Intensity Models [33, 74].

References

1. Acerbi, E., Stella, F.: Continuous time bayesian networks for gene network reconstruction: a comparative study on time course data. In: Proceedings of the 10th International Symposium on Bioinformatics Research and Applications, to appear (2014)
2. Aggarwal, C.C., Han, J., Wang, J., Yu, P.S.: On demand classification of data streams. In: Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '04, pp. 503–508. ACM, New York, NY, USA (2004)
3. Angulo, E., Romero, F.P., García, R., Serrano-Guerrero, J., Olivas, J.A.: An adaptive approach to enhanced traffic signal optimization by using soft-computing techniques. Expert Systems with Applications **38**(3), 2235–2247 (2011)
4. Arnott, R., Small, K.: The economics of traffic congestion. American Scientist **82**(5), 446–455 (1994)
5. Barber, D., Cemgil, A.: Graphical models for time-series. Signal Processing Magazine, IEEE **27**(6), 18–28 (2010)
6. Bradley, P., Fayyad, U., Reina, C.: Scaling clustering algorithms to large databases. pp. 9–15. AAAI Press (1998)
7. Chatterjee, S., Russell, S.J.: Why are dbns sparse? In: Y.W. Teh, D.M. Titterington (eds.) AISTATS, *JMLR Proceedings*, vol. 9, pp. 81–88. JMLR.org (2010)
8. Chickering, D.M., Heckerman, D., Meek, C.: Large-sample learning of bayesian networks is np-hard. The Journal of Machine Learning Research **5**, 1287–1330 (2004)
9. Codecasa, D., Stella, F.: Conditional log-likelihood for continuous time bayesian network classifiers. In: International Workshop NFMCP held at ECML-PKDD 2013 (2013)

10. Codecasa, D., Stella, F.: CTBNCToolkit: Continuous Time Bayesian Network Classifier Toolkit. ArXiv e-prints (2014)
11. Codecasa, D., Stella, F.: Learning continuous time bayesian network classifiers. International Journal of Approximate Reasoning, to appear (2014)
12. Cohn, I., El-Hay, T., Friedman, N., Kupferman, R.: Mean field variational approximation for continuous-time bayesian networks. The Journal of Machine Learning Research **9999**, 2745–2783 (2010)
13. Dacorogna, M.: An introduction to high-frequency finance. AP (2001)
14. Daigle, G., Krueger, G.D., Clark, J.: Tsis: advanced traffic software tools for the user. In: Traffic Congestion and Traffic Safety in the 21st Century: Challenges, Innovations, and Opportunities (1997)
15. Dean, T., Kanazawa, K.: A model for reasoning about persistence and causation. Computational Intelligence **5**(2), 142–150 (1989)
16. Domingos, P., Hulten, G.: Mining high-speed data streams. In: Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '00, pp. 71–80. ACM, New York, NY, USA (2000)
17. El-Hay, T., Cohn, I., Friedman, N., Kupferman, R.: Continuous-time belief propagation. In: J. Fürnkranz, T. Joachims (eds.) Proceedings of the 27th International Conference on Machine Learning (ICML), pp. 343–350. Omnipress, Haifa, Israel (2010)
18. El-Hay, T., Friedman, N., Kupferman, R.: Gibbs sampling in factorized continuous-time markov processes. In: D.A. McAllester, P. Myllym (eds.) Proc. of the 24th Conf. on UAI, pp. 169–178. AUAI (2008)
19. Enright, C.: A probabilistic framework based on mathematical models with application to medical data streams. Ph.D. thesis (2012)
20. Fan, Y., Shelton, C.: Sampling for approximate inference in continuous time bayesian networks. In: 10th Int. Symposium on Artificial Intelligence and Mathematics (2008)
21. Farnstrom, F., Lewis, J., Elkan, C.: Scalability for clustering algorithms revisited. SIGKDD Explor. Newsl. pp. 51–57 (2000). DOI 10.1145/360402.360419. URL http://doi.acm.org/10.1145/360402.360419
22. Favilla, J., Machion, A., Gomide, F.: Fuzzy traffic control: adaptive strategies. In: Fuzzy Systems, 1993., Second IEEE International Conference on, pp. 506–511. IEEE (1993)
23. Felici, G., Rinaldi, G., Sforza, A., Truemper, K.: A logic programming based approach for on-line traffic control. Transportation Research Part C: Emerging Technologies **14**(3), 175–189 (2006)
24. Forster, A., Young, J.: The clinical and cost effectiveness of physiotherapy in the management of elderly people following a stroke. London, UK: The Chartered Society Of Physiotherapy (2002)
25. Fowlkes, E.B., Mallows, C.L.: A method for comparing two hierarchical clusterings. Journal of the American statistical association **78**(383), 553–569 (1983)
26. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Machine Learning **29**(2), 131–163 (1997)
27. Gaber, M.M., Zaslavsky, A., Krishnaswamy, S.: Mining data streams: A review. SIGMOD Rec. **34**(2), 18–26 (2005)
28. Gan, G., Ma, C., Wu, J.: Data clustering: theory, algorithms, and applications, vol. 20. Siam (2007)
29. Gershenson, C.: Self-organizing traffic lights. arXiv preprint nlin/0411066 (2004)
30. Gopalratnam, K., Kautz, H., Weld, D.S.: Extending continuous time bayesian networks. In: Proceedings of the National Conference on Artificial Intelligence, p. 981. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999 (2005)
31. Grossman, D., Domingos, P.: Learning bayesian network classifiers by maximizing conditional likelihood. In: Proc. of the 21st Int. Conf. on Machine Learning, pp. 361–368. ACM (2004)
32. Gualtieri, M., Rigamonti, L., Galeotti, V., Camatini, M.: Toxicity of tire debris extracts on human lung cell line a549. Toxicology in vitro **19**(7), 1001–1008 (2005)
33. Gunawardana, A., Meek, C., Xu, P.: A model for temporal dependencies in event streams. In: Neural Information Processing Systems, pp. 1962–1970. Neural Information Processing Systems Foundation (2011)

34. Haijema, R., van der Wal, J.: An mdp decomposition approach for traffic control at isolated signalized intersections. Probability in the Engineering and Informational Sciences **22**(04), 587–602 (2008)
35. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: On clustering validation techniques. Journal of Intelligent Information Systems **17**(2-3), 107–145 (2001)
36. Halkidi, M., Batistakis, Y., Vazirgiannis, M.: Cluster validity methods: part i. ACM Sigmod Record **31**(2), 40–45 (2002)
37. Hirankitti, V., Krohkaew, J.: An agent approach for intelligent traffic-light control. In: Modelling & Simulation, 2007. AMS'07. First Asia International Conference on, pp. 496–501. IEEE (2007)
38. Hulten, G., Spencer, L., Domingos, P.: Mining time-changing data streams. In: Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '01, pp. 97–106. ACM, New York, NY, USA (2001)
39. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowledge and information systems **7**(3), 358–386 (2005)
40. Koller, D., Friedman, N.: Probabilistic graphical models: principles and techniques. The MIT Press (2009)
41. Kranen, P., Assent, I., Baldauf, C., Seidl, T.: The clustree: Indexing micro-clusters for anytime stream mining. In: Knowledge and Information Systems Journal (Springer KAIS), Volume 29, Issue 2, pp. 249–272. Springer, London (2011)
42. Kwakkel, G., van Peppen, R., Wagenaar, R.C., Dauphinee, S.W., Richards, C., Ashburn, A., Miller, K., Lincoln, N., Partridge, C., Wellwood, I., et al.: Effects of augmented exercise therapy time after stroke a meta-analysis. Stroke **35**(11), 2529–2539 (2004)
43. Lämmer, S., Helbing, D.: Self-control of traffic lights and vehicle flows in urban road networks. Journal of Statistical Mechanics: Theory and Experiment **2008**(04), P04,019 (2008)
44. Law, Y.N., Zaniolo, C.: An adaptive nearest neighbor classification algorithm for data streams. In: Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases, PKDD'05, pp. 108–120. Springer-Verlag, Berlin, Heidelberg (2005)
45. Mantecca, P., Gualtieri, M., Andrioletti, M., Bacchetta, R., Vismara, C., Vailati, G., Camatini, M.: Tire debris organic extract affects¡ i¿ xenopus¡/i¿ development. Environment international **33**(5), 642–648 (2007)
46. Murphy, K., et al.: The bayes net toolbox for matlab. Computing science and statistics **33**(2), 1024–1034 (2001)
47. Nodelman, U., Koller, D., Shelton, C.: Expectation propagation for continuous time bayesian networks. In: Proc. of the 21st Conf. on UAI, pp. 431–440. Edinburgh, Scotland, UK (2005)
48. Nodelman, U., Shelton, C., Koller, D.: Continuous time bayesian networks. In: Proc. of the 18th Conf. on UAI, pp. 378–387. Morgan Kaufmann (2002)
49. Nodelman, U., Shelton, C., Koller, D.: Learning continuous time bayesian networks. In: Proc. of the 19th Conf. on UAI, pp. 451–458. Morgan Kaufmann (2002)
50. Nodelman, U., Shelton, C.R., Koller, D.: Expectation maximization and complex duration distributions for continuous time bayesian networks. CoRR **abs/1207.1402** (2012)
51. Owen, L.E., Zhang, Y., Rao, L., McHale, G.: Street and traffic simulation: traffic flow simulation using corsim. In: Proceedings of the 32nd conference on Winter simulation, pp. 1143–1147. Society for Computer Simulation International (2000)
52. Paolucci, S., Antonucci, G., Grasso, M.G., Morelli, D., Troisi, E., Coiro, P., Bragoni, M.: Early versus delayed inpatient stroke rehabilitation: a matched comparison conducted in italy. Archives of physical medicine and rehabilitation **81**(6), 695–700 (2000)
53. Papageorgiou, M., Diakaki, C., Dinopoulou, V., Kotsialos, A., Wang, Y.: Review of road traffic control strategies. Proceedings of the IEEE **91**(12), 2043–2067 (2003)
54. Park, B., Messer, C.: A genetic algorithm-based signal optimization program for oversaturated intersections. In: Towards the new horizon together. Proceeding of the 5th world congress of intelligent transport system., 1026. Seoul, Korea (1998)
55. Rabiner, L.: A tutorial on hidden markov models and selected applications in speech recognition. Proc. of the IEEE **77**(2), 257–286 (1989)

56. Rajaram, S., Graepel, T., Herbrich, R.: Poisson-networks: A model for structured point processes. In: Proc. of the 10th Int. Workshop on Artificial Intelligence and Statistics, pp. 277–284 (2005)
57. Rand, W.M.: Objective criteria for the evaluation of clustering methods. Journal of the American Statistical association **66**(336), 846–850 (1971)
58. Rao, V., Teh, Y.W.: Fast mcmc sampling for markov jump processes and continuous time bayesian networks. arXiv preprint arXiv:1202.3760 (2012)
59. Robertson, D.I.: " transyt" method for area traffic control. Traffic Engineering & Control **11**(6) (1969)
60. Robertson, D.I., Bretherton, R.D.: Optimizing networks of traffic signals in real time-the scoot method. Vehicular Technology, IEEE Transactions on **40**(1), 11–15 (1991)
61. Saria, S., Nodelman, U., Koller, D.: Reasoning at the right time granularity. In: UAI, pp. 326–334 (2007)
62. Silva, J.A., Faria, E.R., Barros, R.C., Hruschka, E.R., Carvalho, A.C.P.L.F.d., Gama, J.a.: Data stream clustering: A survey. ACM Comput. Surv. **46**(1), 13:1–13:31 (2013). DOI 10.1145/2522968.2522981. URL http://doi.acm.org/10.1145/2522968.2522981
63. Simma, A., Goldszmidt, M., MacCormick, J., Barham, P., Black, R., Isaacs, R., Mortier, R.: Ct-nor: Representing and reasoning about events in continuous time. In: Proc. of the 24th Conf. on UAI, pp. 484–493. AUAI (2008)
64. Simma, A., Jordan, M.: Modeling events with cascades of poisson processes. In: Proc. of the 26th Conf. on UAI, pp. 546–555. AUAI (2010)
65. Sims, A.G., Dobinson, K.: The sydney coordinated adaptive traffic (scat) system philosophy and benefits. Vehicular Technology, IEEE Transactions on **29**(2), 130–137 (1980)
66. Spall, J.C., Chin, D.C.: Traffic-responsive signal timing for system-wide traffic control. Transportation Research Part C: Emerging Technologies **5**(3-4), 153–163 (1997)
67. Stella, F., Amer, Y.: Continuous time bayesian network classifiers. Journal of Biomedical Informatics **45**(6), 1108–1119 (2012)
68. Stella, F., Viganò, V., Bogni, D., Benzoni, M.: An integrated forecasting and regularization framework for light rail transit systems. Journal of Intelligent Transportation Systems **10**(2), 59–73 (2006)
69. Thorpe, T.L., Anderson, C.W.: Tra c light control using sarsa with three state representations. Tech. rep., Citeseer (1996)
70. Tormene, P., Giorgino, T., Quaglini, S., Stefanelli, M.: Matching incomplete time series with dynamic time warping: an algorithm and an application to post-stroke rehabilitation. Artificial Intelligence in Medicine **45**(1), 11–34 (2009)
71. Truccolo, W., Eden, U., Fellows, M., Donoghue, J., Brown, E.: A point process framework for relating neural spiking activity to spiking history, neural ensemble, and extrinsic covariate effects. Journal of neurophysiology **93**(2), 1074–1089 (2005)
72. Villa, S., Stella, F.: A continuous time bayesian network classifier for intraday fx prediction. Quantitative Finance pp. 1–14 (2014). DOI 10.1080/14697688.2014.906811
73. Voit, E.: A First Course in Systems Biology. Garland Science: NY (2012)
74. Weiss, J.C., Page, D.: Forest-based point process for event prediction from electronic health records. In: Machine Learning and Knowledge Discovery in Databases, pp. 547–562. Springer (2013)
75. Weiss, J.C., Page, D.: Forest-based point processes for event prediction from electronic health records. Machine Learning and Knowledge Discovery in Databases **8190**, 547–562 (2013)
76. Wiering, M.: Multi-agent reinforcement learning for traffic light control (2000)
77. Xu, R., Wunsch, D.: Clustering, vol. 10. Wiley. com (2008)
78. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. ACM Computing Surveys **38**(4), 1–45 (2006). DOI 10.1145/1177352.1177355
79. Yu, X.H., Recker, W.W.: Stochastic adaptive control model for traffic signal systems. Transportation Research Part C: Emerging Technologies **14**(4), 263–282 (2006)
80. Zhou, A., Cao, F., Qian, W., Jin, C.: Tracking clusters in evolving data streams over sliding windows. Knowl. Inf. Syst. **15**(2), 181–214 (2008)