Journal of Biomedical Informatics 45 (2012) 1108–1119

Contents lists available at SciVerse ScienceDirect

Journal of Biomedical Informatics

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

Continuous time Bayesian network classifiers

F. Stella *, Y. Amer

Department of Informatics, Systems and Communication, Università degli Studi di Milano-Bicocca, Viale Sarca 336, 20126 Milano, Italy

ARTICLE INFO

Article history:
Received 17 October 2011
Accepted 15 July 2012
Available online 28 July 2012

Keywords:
Bayesian classifiers
Continuous time
Multivariate trajectory

ABSTRACT

The class of continuous time Bayesian network classifiers is defined; it solves the problem of supervised classification on multivariate trajectories evolving in continuous time. The trajectory consists of the values of discrete attributes that are measured in continuous time, while the predicted class is expected to occur in the future. Two instances from this class, namely the continuous time naive Bayes classifier and the continuous time tree augmented naive Bayes classifier, are introduced and analyzed. They implement a trade-off between computational complexity and classification accuracy. Learning and inference for the class of continuous time Bayesian network classifiers are addressed, in the case where complete data are available. A learning algorithm for the continuous time naive Bayes classifier and an exact inference algorithm for the class of continuous time Bayesian network classifiers are described. The performance of the continuous time naive Bayes classifier is assessed in the case where real-time feedback to neurological patients undergoing motor rehabilitation must be provided.

1. Introduction

Temporal data classification is concerned with databases containing measurements over a period of time in history, while the predicted class is expected to occur in the future.

Hidden Markov models (HMMs) [1], dynamic Bayesian networks (DBNs) [2] and dynamic network models [3], which have been used to solve *dynamic supervised classification problems* [4–9], represent time through discrete time points. HMMs have been extended to dynamic naive Bayesian classifiers (DNBCs), while a methodology to automatically learn these models from data has been described in [10].

An extension of Bayesian network classifiers (BNCs) [11], called temporal Bayesian classifiers (TBCs), adopts a more compact representation than DBNs. TBCs augment the BNCs with a root node, other than the class one, whose state is associated with discrete time points. They have been used to diagnose muscular dystrophy from gene expression data [12] and for spatio-temporal understanding of the visual field deterioration [13].

To the best of the authors' knowledge no supervised classification approaches based on probabilistic graphical models have been described which explicitly model time. Indeed, HMMs, DBNs, DNBCs and TBCs do not model time explicitly, while the importance to model and reason with time is well understood and recognized in several domains [14,15].

Recently, an alternative to HMMs, DBNs, DNBCs and TBCs, which extends BNs [16,17] from static to dynamic models by explicitly modeling time, has been offered by continuous time Bayesian networks [18]. This network model overcomes the main limitations of HMMs, DBNs, DNBCs and TBCs by explicitly representing temporal dynamics and allows us to recover the probability distribution over time when specific events occur.

In this paper the continuous time Bayesian networks are translated into a new class of supervised classification models, the class of continuous time Bayesian network classifiers. The paper makes the following contributions:

- Defines a new class of supervised classifiers; namely the class of continuous time Bayesian network classifiers. Two parsimonious instances from this class, i.e. the continuous time naive Bayes classifier and the continuous time tree augmented naive Bayes classifier, are introduced.
- Develops an exact algorithm for making inference on all instances from the class of continuous time Bayesian network classifiers.
- Compares the continuous time naive Bayes classifier with two algorithms, namely dynamic time warping and open-end dynamic time warping which is a state-of-the art algorithm for real-time feedback to neurological patients undergoing motor rehabilitation.

The rest of the paper is organized as follows. Section 2 gives the basics of continuous time Bayesian networks together with an overview on learning and inference. In Section 3 the class of continuous time Bayesian network classifiers is introduced, together with their specializations in the continuous time naive Bayes classifier and in the continuous time tree augmented naive Bayes

* Corresponding author.
E-mail address: stella@disco.unimib.it (F. Stella).

1532-0464/$ - see front matter © 2012 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.jbi.2012.07.002

classifier. The effectiveness of the continuous time naive Bayes classifier is evaluated by comparing its classification accuracy against the accuracy achieved by the dynamic time warping algorithm and its real-time open-end version proposed in [19], specifically designed to help patients in their post-stroke rehabilitation. Finally, Section 5 is devoted to the conclusions.

2. Continuous time Bayesian networks

Dynamic Bayesian networks (DBNs) [2] are the standard extension of BNs [20] when dealing with dynamical systems. They do not model time explicitly while they discretize it to represent a dynamical system through several time slices.

However, "*since DBNs slice time into fixed increments, one must always propagate the joint distribution over the variables at the same rate*" [18]. 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 main limitations of DBNs by explicitly representing temporal dynamics and thus allow us to recover the probability distribution over time when specific events occur. They have been used to model the presence of people at their computers, together with the specific application they are using (e.g., email, word processing, web browsing, etc.) [21], for dynamical systems reliability modeling and analysis [22], for network intrusion detection [23], for modeling social networks [24] and cardiogenic heart failure [25].

A continuous time Bayesian network (CTBN) is a graphical model whose nodes are associated with random variables and whose state evolves continuously over time. As a consequence the evolution of each variable depends on the state of its parents in the graph. A CTBN consists of two main components: (i) an initial probability distribution and (ii) the dynamics which rule the evolution over time of the probability distribution associated with the CBTN.

Definition 1 18 Continuous time Bayesian network. Let X be a set of random variables $X_1, X_2, \ldots, X_N$. Each X_n has a finite domain of values $Val(X_n) = \{x_1, x_2, \ldots, x_I\}$. A continuous time Bayesian network $\aleph$ over X consists of two components: the first is an initial distribution $P^0_{\mathbf{X}}$, specified as a Bayesian network $\mathcal{B}$ over X. The second is a continuous transition model, specified as

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

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

$$\mathbf{Q}^{pa(x_n)}_{X_n} = \begin{bmatrix} -q^{pa(x_n)}_{x_1} & q^{pa(x_n)}_{x_1 x_2} & . & q^{pa(x_n)}_{x_1 x_I} \\ q^{pa(x_n)}_{x_2 x_1} & -q^{pa(x_n)}_{x_2} & . & q^{pa(x_n)}_{x_2 x_I} \\ . & . & . & . \\ q^{pa(x_n)}_{x_I x_1} & q^{pa(x_n)}_{x_I x_2} & . & -q^{pa(x_n)}_{x_I} \end{bmatrix},$$

for each instantiation $pa(x_n)$ of the parents $pa(X_n)$ of node X_n, where $q^{pa(x_n)}_{x_i} = \sum_{x_j \neq x_i} q^{pa(x_n)}_{x_i x_j}$, can be interpreted as the *instantaneous probability* to leave x_i for a specific instantiation $pa(x_n)$ of $pa(X_n)$, while $q^{pa(x_n)}_{x_i x_j}$ can be interpreted as the instantaneous probability to transition from x_i to x_j for a specific instantiation $pa(x_n)$ of $pa(X_n)$.

CTBNs allow *point evidence* and *continuous evidence* while HMMs, DBNs, DNBCs and TBCs allow only *point evidence*. Point evidence is an observation of the value x of a variable X_n at a particular instant in time, i.e. $X_n(t) = x$ while continuous evidence provides the value of a variable throughout an entire interval, which we take to be a half-closed interval $[t_1, t_2)$. CTBNs are instantiated with an evidence sequence which will be called *evidence stream*. This concept is better clarified with the following definitions.

Definition 2 (*J-time-stream*). A J-time-stream, over the left-closed time interval $[0,T)$, is a partitioning into J left-closed intervals $[0,t_1); [t_1,t_2); \ldots; [t_{J-1},T)$.

Definition 3 (*J-evidence-stream*). Given a CTBN $\aleph$, consisting of N nodes, and a J-time-stream $[0,t_1); [t_1,t_2); \ldots; [t_{J-1},T)$, a J-evidence-stream is the set of joint instantiations $\mathbf{X} = \mathbf{x}$ for any subset of random variables X_n, $n = 1, 2, \ldots, N$ associated with each of the J time segments. A J-evidence-stream will be referred to as $(\mathbf{X}^1 = \mathbf{x}^1, \mathbf{X}^2 = \mathbf{x}^2, \ldots, \mathbf{X}^J = \mathbf{x}^J)$ or for short as $(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)$.

A *J-evidence-stream* $(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)$ is said to be *fully observed* in the case where the state of all the variables X_n is known along the whole time interval $[0,T)$. A *J-evidence-stream* which is not fully observed is said to be *partially observed*. Given a CTBN and a *J-evidence-stream* (fully or partially observed), inference is concerned with the computation of the posterior for those variables whose state is unknown.

Exact inference in CTBNs is NP-hard, and thus several approximate algorithms have been proposed; *expectation propagation* (EP) [26], EP-based algorithm using a flexible cluster graph architecture [27], importance sampling [28] and Gibbs sampling [29].

The problem of learning a CTBN from data $\mathcal{D}$ has been addressed in [30] as the problem of finding the structure $\mathcal{G}$ which maximizes the following score:

$$\mathbf{score}_{\aleph}(\mathcal{G} : \mathcal{D}) = \ln P(\mathcal{D}|\mathcal{G}) + \ln P(\mathcal{G}). \tag{1}$$

This is an optimization problem over possible CTBN structures whose search space is significantly simpler than that of BNs or DBNs. While it is known that learning the optimal structure of a BN is NP-hard, the same does not hold true in the context of CTBN learning where 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.

3. Continuous time Bayesian network classifiers

The CTBN model can be exploited to define a new class of supervised classification models which explicitly represent the evolution in continuous time of the set of random variables X_n, $n = 1, 2, \ldots, N$. To define this new class of supervised classification models, which will be called the class of continuous time Bayesian network classifiers (CTBNCs), an additional node Y associated with the class is required. The continuous time Bayesian network classifier (CTBNC) is defined as follows.

Definition 4 (*Continuous time Bayesian network classifier*). A continuous time Bayesian network classifier is a pair $\mathcal{C} = \{\aleph, P(Y)\}$ where $\aleph$ is a CTBN model with attribute nodes $X_1, X_2, \ldots, X_N$, class node Y with marginal probability $P(Y)$ on states $Val(Y) = \{y_1, y_2, \ldots, y_K\}$, $\mathcal{G}$ is the graph of the CTBNC, such that the following conditions hold:

- $\mathcal{G}$ is connected.
- $pa(Y) = \emptyset$, the class variable Y is associated with a root node.
- Y is fully specified by $P(Y)$ and does not depend on time.

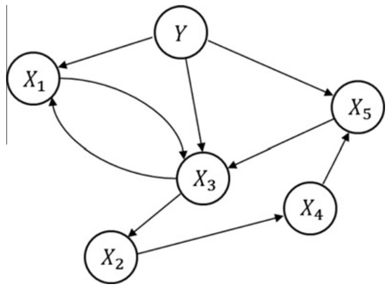

Fig. 1. Continuous time Bayesian network classifier; five attribute nodes $X_1, \ldots, X_5$ and the class Y.

A CTBNC instance, with attributes X_1, X_2, X_3, X_4, X_5 and class Y, is depicted in Fig. 1. The class variable Y is associated with a root node, i.e. a node with no parent nodes. This model contains two cycles; the first one involves the following attributes X_3, X_2, X_4 and X_5 while the second one involves the attribute X_1 which causes the attribute X_3 which in turn causes X_1.

Several considerations concerning the exploitation of the structure of the graph $\mathcal{G}$ of the CTBNC in the case where the classification task is performed on a *fully observed J-evidence-stream* can be made. In such an evidential setting, the only unobserved random variable is the class variable Y; thus it is possible to fruitfully and conditionally exploit independence relationships between random variables as it happens in ordinary BNs.

Given a data set $\mathcal{D}$ consisting of *fully observed evidence streams*, a CTBNC can be learnt by maximizing the score function $\mathbf{score}_{\aleph}(\mathcal{G} : \mathcal{D})$ (1) subjected to the constraints listed in Definition 4. Exact learning requires to set in advance the maximum number of parents L for the nodes $X_1, X_2, \ldots, X_N$ [30]. In the case where L is not small a considerable computational effort is required to find the optimal graph structure $\mathcal{G}$, i.e. the one which maximises the score function $\mathbf{score}_{\aleph}(\mathcal{G} : \mathcal{D})$ (1). In such a setting, the set of constraints due to Definition 4 does not help very much to reduce the time required to find the optimal solution. Therefore, we have to resort to hill-climbing optimization procedures to find an approximate solution to the considered optimization problem.

Following the motivations presented in [11] concerning the extension from BNs to BNCs and subsequently from BNCs to TAN, we propose the following structurally constrained classifier instances belonging to the class of CTBNCs: the continuous time naive Bayes (CTNB) classifier and the continuous time tree augmented naive Bayes (CTTANB) classifier. The CTNB classifier (Fig. 2) is such that the class variable Y is a root node which is the only parent for the attributes X_n, $n = 1, 2, \ldots, N$.

Definition 5 (*Continuous time naive Bayes classifier*). A continuous time naive Bayes classifier is a continuous time Bayesian network classifier $\mathcal{C} = \{\aleph, P(Y)\}$ such that $pa(X_n) = Y$, $n = 1, 2, \ldots, N$.

Definition 6 (*Continuous time tree augmented naive Bayes classifier*). A continuous time tree augmented naive Bayes classifier is a continuous time Bayesian network classifier $\mathcal{C} = \{\aleph, P(Y)\}$ such that the following conditions hold:

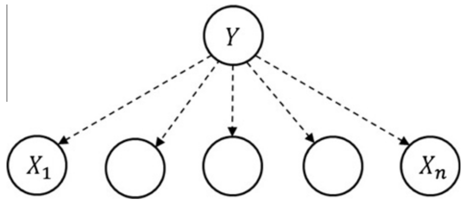

Fig. 2. Continuous time naive Bayes classifier.

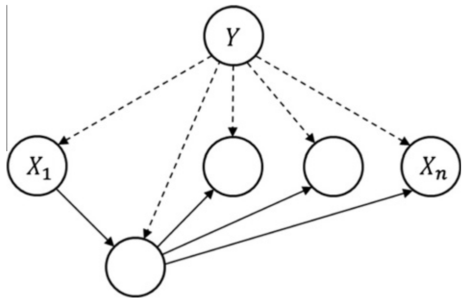

Fig. 3. Continuous time tree augmented naive Bayes classifier: if the class variable Y is removed, the remaining nodes form a tree.

- $Y \in pa(X_n)$, $n = 1, 2, \ldots, N$.
- the attribute nodes X_n, $n = 1, 2, \ldots, N$, form a tree, $\exists j: |pa(X_j)| = 1$ while for $i \neq j$, $|pa(X_i)| = 2$.

The CTTANB classifier (Fig. 3) is such that the class variable Y is a root node while each attribute X_n, $n = 1, 2, \ldots, N$ is constrained to have as parents the class variable Y and at most one of the remaining attributes.

It is worthwhile to mention that the learning problem for the CTNB classifier reduces to the estimation of the CIMs parameters, while for the CTTANB classifier the exact solution can be found with little computational effort. Therefore, the CTNB and CTTANB classifiers are interesting supervised classification models from the class of CTBNCs which implement trade-off between computational complexity and classification performance.

A CTBNC $\mathcal{C} = \{\aleph, P(Y)\}$ when presented with a *fully observed J-evidence-stream* $(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)$ classifies it according to the maximum aposteriori rule. The posterior probability $P(Y|(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J))$ is computed for all the states of the class variable Y and the CTBNC classifies a *fully observed J-evidence-stream* with the state $y \in Val(Y)$ maximizing the posterior probability:

$$P(Y|(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)) = \frac{P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)|Y)P(Y)}{P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J))}, \tag{2}$$

where $P(Y)$ is the marginal probability associated with the class variable Y while the terms $P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)|Y)$ and $P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J))$ deserve a more detailed description. $P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J))$ represents the probability of the evidence which in this particular setting will be called *probability of the fully observed J-evidence-stream*. This quantity, similarly to what happens in the case where BNCs models are used, is not required to implement the maximum aposteriori rule. Therefore, it is possible to write the following proportionality relationship:

$$P(Y|(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)) \propto P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)|Y)P(Y). \tag{3}$$

The term $P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)|Y)$ is the likelihood of the *J-evidence-stream*, given the class variable. This term, which will be referred to as *temporal likelihood*, is fundamental to implement the maximum aposteriori classification rule. The temporal likelihood, can be expressed as follows:

$$P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)|Y) = \prod_{j=1}^{J} P(\mathbf{x}^j|Y)P(\mathbf{x}^{j+1}|\mathbf{x}^j, Y), \tag{4}$$

where $P(\mathbf{x}^j|Y)$ represents the probability that $\boldsymbol{X}$ stays in state $\mathbf{x}^j$ during the time interval $[t_{j-1}, t_j)$ given the class variable Y while the term $P(\mathbf{x}^{j+1}|\mathbf{x}^j, Y)$ represents the probability that $\mathbf{X}$ transitions from state $\mathbf{x}^j$ to state $\mathbf{x}^{j+1}$ at time t_j given the class variable Y (to ensure consistency we set $P(\mathbf{x}^{J+1}|\mathbf{x}^J, Y)$=1).

In the case where a model from the class of CTBNCs is considered, the term $P(\mathbf{x}^j|Y)$ can be written as follows:

$$P(\mathbf{x}^j|Y) = \prod_{n=1}^{N} exp\left(-q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})\right), \tag{5}$$

where $q_{x_n^j}^{pa(X_n)}$ is the value of the parameter associated with the state x_n^j in which the variable X_n stays during the jth time segment $[t_{j-1},t_j)$ given the state of its parents $pa(X_n)$ during the same time segment (we recall that $t_0 = 0$ and $t_J = T$). Under the same assumption, the term $P(\mathbf{x}^{j+1}|\mathbf{x}^j,Y)$ can be written as follows:

$$P(\mathbf{x}^{j+1}|\mathbf{x}^j, Y) = \prod_{n=1}^{N} P(x_n^{j+1}|x_n^j, Y), \tag{6}$$

where

$$P(x_n^{j+1}|x_n^j, Y) = \begin{cases} 1 - exp\left(-q_{x_n^j x_n^{j+1}}^{pa(X_n)}\epsilon\right) & \text{if } x_n^j \neq x_n^{j+1} \\ 1 & \text{otherwise} \end{cases}, \tag{7}$$

while

- $q_{x_n^j x_n^{j+1}}^{pa(X_n)}$ is the value of the parameter associated with the transition from state x_n^j, in which the variable X_n was during the jth time segment $[t_{j-1},t_j)$, to state x_n^{j+1}, in which the variable X_n will be during the $(j+1)$th time segment $[t_j,t_{j+1})$, given the state of its parents $pa(X_n)$ during the jth time segment,
- ϵ is a small positive number, needed to make inference on point evidence.

The Eqs. (5) and (6) can be combined to write the following:

$$P(\mathbf{x}^j|Y)P(\mathbf{x}^{j+1}|\mathbf{x}^j, Y) = \prod_{n=1}^{N} exp\left(-q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})\right)P(x_n^{j+1}|x_n^j, Y). \tag{8}$$

Term $P\left(x_n^{j+1}|x_n^j, Y\right)$, defined according to (7), represents the probability that the random vector $\boldsymbol{X}$ transitions from state $\mathbf{x}^j$ to state $\mathbf{x}^{j+1}$ given the value of the class variable Y.

CTBNs imply that at each time step t_j, $j = 1, 2, \ldots, J$, exactly one component X_n of the random vector $\boldsymbol{X}$ is subject to state transition. Therefore, the formula (7), in the case where the parameter $\epsilon \to 0$, computes the probability of the instantaneous transitioning of the random variable X_n from the state x_n^j to the state x_n^{j+1}, while the other components of the random vector $\boldsymbol{X}$, i.e. X_r, $r \neq n$, do not change their state. The substitution of (8) into (4) allows to write the following expression for the temporal likelihood:

$$P((\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)|Y) = \prod_{j=1}^{J}\prod_{n=1}^{N} exp\left(-q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})\right)P(x_n^{j+1}|x_n^j, Y), \tag{9}$$

while by substituting (9) into (3) it is possible to write:

$$P(Y|(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)) \propto P(Y)\prod_{j=1}^{J}\prod_{n=1}^{N} exp\left(-q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})\right)P(x_n^{j+1}|x_n^j, Y). \tag{10}$$

Therefore, given a CTBNC $\mathcal{C} = \{\aleph, P(Y)\}$ and a *fully observed J-evidence-stream* $(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J)$ the maximum aposteriori classification rule selects the class $y^* \in Val(Y)$ as follows:

$$y^* = \arg\max_{Y \in Val(Y)} P(Y)\prod_{j=1}^{J}\prod_{n=1}^{N} exp\left(-q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})\right)P\left(x_n^{j+1}|x_n^j, Y\right). \tag{11}$$

Learning and inference algorithms for CTBNCs are presented and described in detail in the Appendix.

4. Post-stroke rehabilitation

Post-stroke rehabilitation is known to be a challenging problem in the clinical course of patients affected by an acute neurological event. The main obstacles with the rehabilitation therapy are: the post-event depression affecting the patient and the limited availability of rehabilitation structures. It is known that when the patient undergoes the therapy early on, the quality and speed of recovery are increased while achieving a cost-effective patient treatment. Furthermore, it is usual practice to keep patients in the acute ward, which is extremely costly, until a place becomes available in rehabilitation structures. This is why it is important to encourage the patients' autonomy along their path to recovery.

Table 1
Description of the seven types of motor rehabilitation exercises.

i	Description of the exercise type
1	Abduction–adduction of the upper limb on a frontal pane
2	Abduction–adduction of upper limb on a sagittal plane
3	External rotation of the forearm
4	Flexion–extension of the elbow
5	Pronation–supination of the forearm
6	Functional activity: eating
7	Functional activity: combing

Table 2
Description of the six modes of execution analyzed for each exercise.

j	Description	Correctness	Speed
1	Reference	Correct	Slow
2	Reference	Correct	Average
3	Reference	Correct	Fast
4	Movement too small	Incorrect	Slow
5	Typical compensatory action (first)	Incorrect	Average
6	Typical compensatory action (second)	Incorrect	Fast

Table 3
Summary of the average values of accuracy achieved by BASELINE, DTW, OE-DTW and CTNB on the seven types of motor rehabilitation exercises for the *prediction of the execution correctness* (2-class), *recognition of the execution correctness versus error type* (4-class) and *prediction of the execution mode* (6-class) experiments. Average values are related to complete trajectories (*Complete*) and to half trajectories (*Half*). In bold the best accuracy for each classification problem, while the 90% confidence interval [31] is reported in brackets. The asterisk (*) is associated with a classifier achieving an accuracy value which is better than those achieved by the other classifiers on the same classification problem, a *Z*-test with 95% confidence was used [32,33].

	Algorithm	2-class	4-class	6-class
Complete	BASELINE	0.500	0.500	~0.167
	DTW	**0.992** (0.985-0.996)	**0.979** (0.969-0.986)	0.877 (0.857-0.894)
	OE-DTW	0.987 (0.979-0.992)	0.974 (0.963-0.982)	0.873 (0.853-0.891)
	CTNB	0.988 (0.980-0.993)	0.975 (0.964-0.982)	**0.930(*)** (0.914-0.943)
Half	DTW	0.771 (0.746-0.794)	0.619 (0.591-0.646)	0.274 (0.249-0.300)
	OE-DTW	0.952 (0.938-0.963)	0.933 (0.917-0.946)	0.836 (0.814-0.856)
	CTNB	**0.975(*)** (0.964-0.982)	**0.939** (0.924-0.951)	**0.886(*)** (0.866-0.903)

In [19] the authors rely on the concept of providing the patient with immediate feedback for rewarding and/or improving his/her performance with self-rehabilitation devices. They describe an algorithm which processes patient's motion signals to check the correctness of the execution of different types of rehabilitation movement. The algorithm detects arm movements in real-time from a sensorized shirt consisting of 29 sensing segments distributed over the arms, forearms and shoulders.

The problem of recognizing the quality of the movement by using the time series, originating from the sensors, arises because no direct correspondence between the amount of fabric stretch of different parts of the garment and biomechanical parameters is known. Furthermore, sensors' readings are highly correlated and in general redundant. In addition, sensors are affected by various sources of uncertainty which make it complex to compute limb positions from sensors' input. Finally, measurements suffer from non-linearities of the sensor response to strain, difference in body build and placement of the garment, even in the case of different sessions of the same patient. Therefore, the considered problem, *posture classification from strain readings* has been tackled by machine learning algorithms. These models only classify instantaneous sensor readings, thus failing to take into account the time-dependent behavior of exercises and sensors response.

The reference dataset used for evaluating the effectiveness of the CTNB classifier is described in [19]. It consists of time series associated with the execution of seven types of motor rehabilitation exercises including deficit-correlated and functional movements, at different speeds, while knowing when they have been performed correctly or not. The exercises have been performed according to a given protocol and indexed by the *sijk* tuple where s refers to the acquisition session ($s = 1,\dots,4$), i to exercise type ($i = 1,\dots,7$) (Table 1), j to mode of execution ($j = 1,\dots,6$) (Table 2) while k to repetitions ($k = 1,\dots,5$). The sensor acquisition protocol includes exercises corresponding to all possible combinations of the *sijk* tuple for a total number of $4 \times 7 \times 6 \times 5 = 840$ multivariate trajectories.

CTBNCs are designed for discrete attributes and thus the trajectories have been discretized, i.e. the range for each attribute has been divided into intervals according to different quantile values. No other pre-processing activity has been performed on the dataset. Furthermore, in all numerical experiments the parameter ϵ of Algorithm 2 has been set to 10^{-6} while no probability threshold has been used to classify the trajectories, i.e. each trajectory is classified by the maximum posterior probability class.

To compare the CTBNCs with the dynamic time warping (DTW) algorithm proposed in [19] the same experimental setting has been

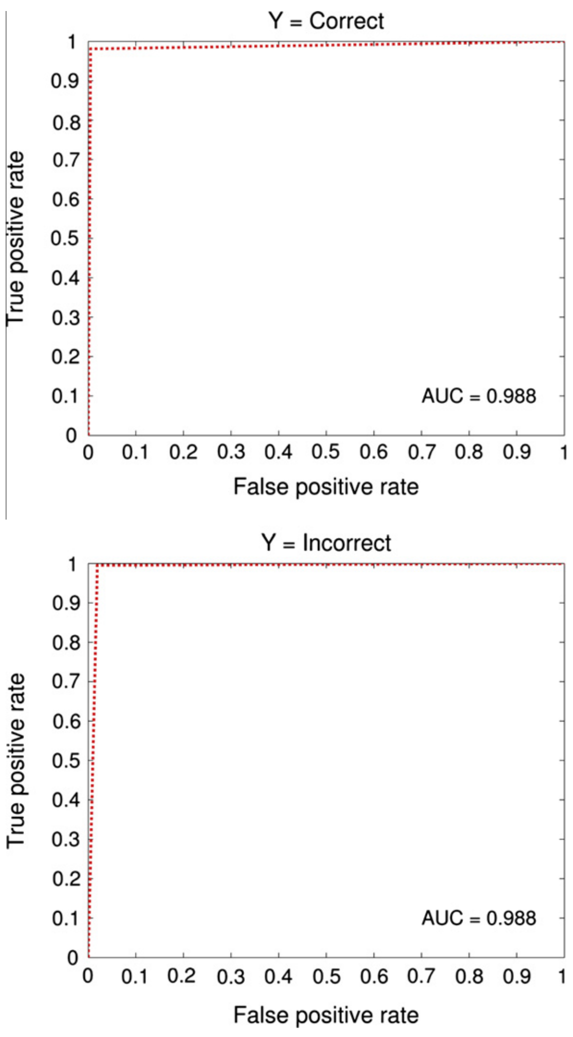

Fig. 4. ROC of the CTNB classifier for the *prediction of the execution correctness* experiment in the case where the full time series data is used (2-class, complete).

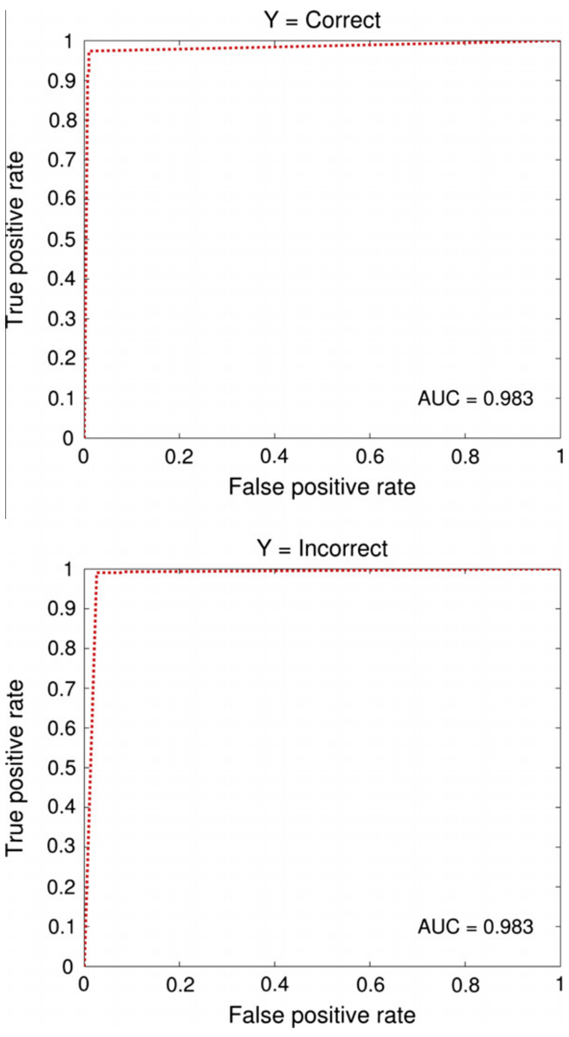

Fig. 5. ROC of the CTNB classifier for the *prediction of the execution correctness* experiment in the case where the first half of each trajectory is used (2-class, half).

adopted. We divided the dataset into seven subsets indexed by *i* and consisting of 120 trajectories.

The estimation of the accuracy relies on leave one out cross-validation, each of the 120 trajectories is classified after the learning of the CTBNC by using the remaining 119 trajectories. Three types of numerical experiments have been performed:

- prediction of the execution correctness,
- recognition of the execution correctness versus error type,
- prediction of the execution mode.

The *prediction of the execution correctness* experiment is formulated as a binary classification problem and aims to discern whether an exercise has been correctly performed or not. The class of correct movements is associated with the execution modes indexed by $j = 1, 2, 3$, thus 420 trajectories are associated with correct movements (Y = *Correct*), while $j = 4, 5, 6$ indices are associated with the incorrect movements (Table 2), thus 420 trajectories are associated with incorrect movements (Y = *Incorrect*). The *prediction of the execution correctness* experiment is a 2-class classification problem where the marginal probability of the two classes is the same.

The *recognition of the execution correctness versus error type* experiment is devoted to establishing whether the analyzed movement was correctly performed or not, and in case it was not to predict the error type. Thus, the execution modes in Table 2 indexed with $j = 1, 2, 3$, consisting of 420 trajectories, are assumed to be correct (Y = *Reference*), while the remaining execution modes, namely *Movement too small* ($j = 4$), *Typical compensatory action (first)* ($j = 5$) and *Typical compensatory action (second)* ($j = 6$) are three error types, each consisting of 140 trajectories. The *recognition of the execution correctness versus error type* experiment is a 4-class classification problem.

It is worthwhile to mention that the marginal probability of the class of correct movements (Y = *Reference*) is equal to *0.5* while the marginal probability for each of the remaining three classes (Y = *Movement too small*, Y = *Typical compensatory action (first)* and Y = *Typical compensatory action (second)*) is $\sim$0.167.

The *prediction of the execution mode* experiment is a 6-class classification problem where each class has marginal probability $\sim$

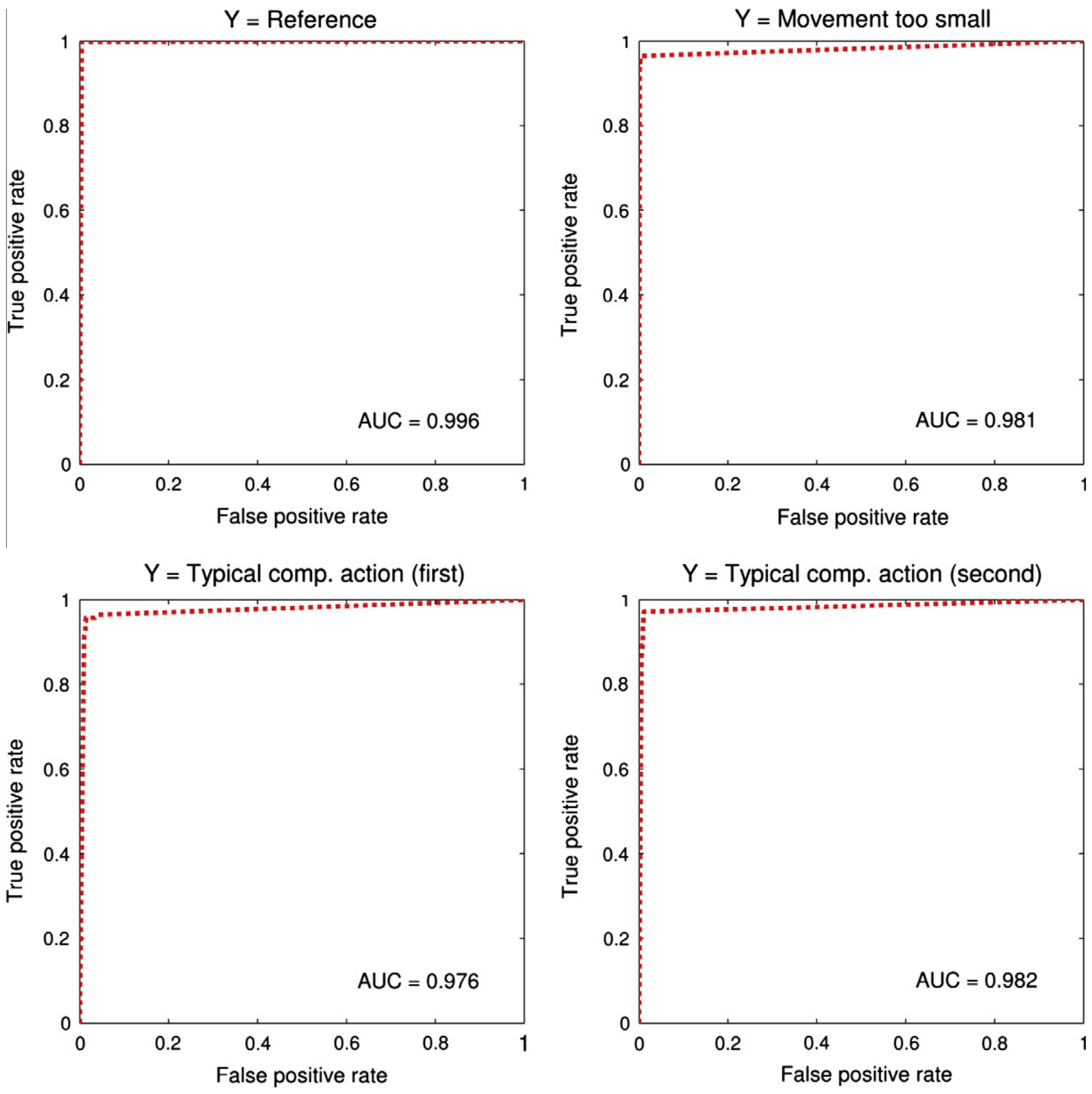

Fig. 6. ROC of the CTNB classifier for the *recognition of execution correctness versus error type* experiment in the case where the full time series data is used (4-class, complete).

0.167. It aims to separate the following six classes (execution modes); *Reference-Slow*, *Reference-Average*, *Reference-Fast*, *Movement too small*, *Typical compensatory action (first)* and *Typical compensatory action (second)* as listed in Table 2.

The average values of the accuracy achieved for the seven types of motor rehabilitation exercise are summarized in Table 3.

The first four lines of Table 3 concern the performance of the BASELINE classifier (the model which classifies each trajectory by the apriori most frequent class), the original DTW algorithm, the open-end DTW (OE-DTW) described in [19] and the CTNB classifier, when they are applied to the full time series (*Complete*). Following [19], the second four lines report the performance of BASELINE, DTW, OE-DTW and CTNB when queried with the first half of each trajectory (*Half*). In this setting only the first half of each trajectory was used. Performance values are related to the 2-class, 4-class and 6-class problem formulations, while the best performance achieved for each classification problem is in bold to make it easier to read-off the results from Table 3. Furthermore, 90% confidence intervals are reported in brackets.

In the *Complete* setting, i.e. when the full time series are used, the best accuracy for the 2-class problem and for the 4-class problem is achieved by the original DTW algorithm. Though the accuracies of the OE-DTW algorithm and the CTNB classifier are slightly worse than the ones of the DTW algorithm, they are not significantly different from each other. The best accuracy for the 6-class is achieved by the CTNB classifier, while the accuracy values achieved by the DTW and OE-DTW algorithms do not significantly differ from each other. However, the accuracy achieved by the CTNB classifier is significantly better, at the 95% confidence, than the performance achieved by the DTW and OE-DTW algorithms.

In the *Half* setting, i.e. when the first half time series are used, the best accuracy is always achieved by the CTNB classifier which uniformly outperforms the DTW and the OE-DTW algorithms. It is worthwhile to mention that for the 2-class and the 6-class problems the CTNB classifier achieved an accuracy which is significantly better, at the 95% confidence, than the performance achieved by the DTW and OE-DTW algorithms.

ROC curves for 2-class, 4-class and 6-class experiments, complete and half, are depicted in Figs. 4–9.

Figs. 10 and 11 depict the evolution along time of the posterior probability of class *Y* as computed by the CTNB classifier.

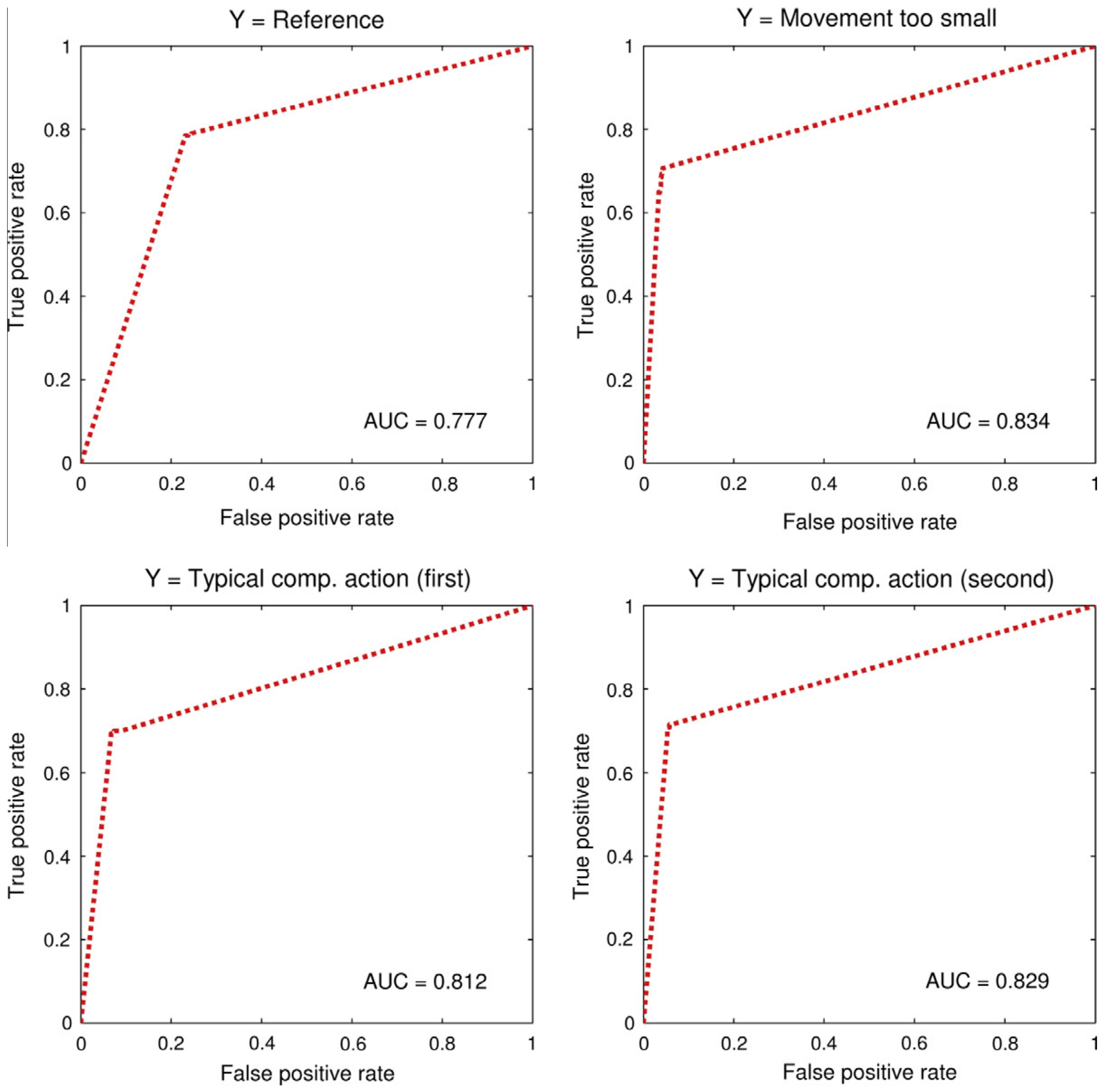

Fig. 7. ROC of the CTNB classifier for the *recognition of execution correctness versus error type* experiment in the case where the first half of each trajectory is used (4-class, half).

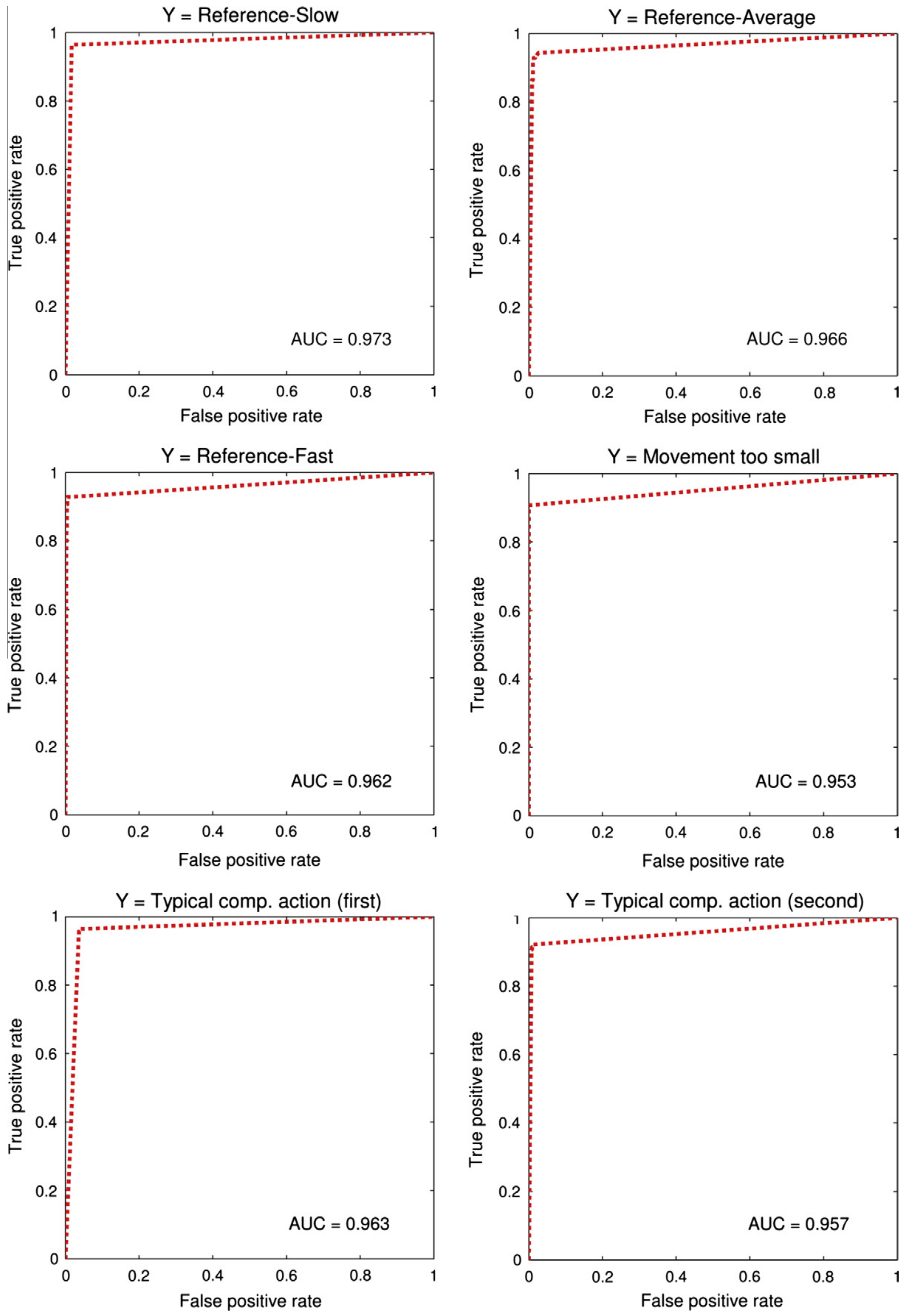

Fig. 8. ROC of the CTNB classifier for the *prediction of the execution mode* experiment in the case where the full time series data is used (6-class, complete).

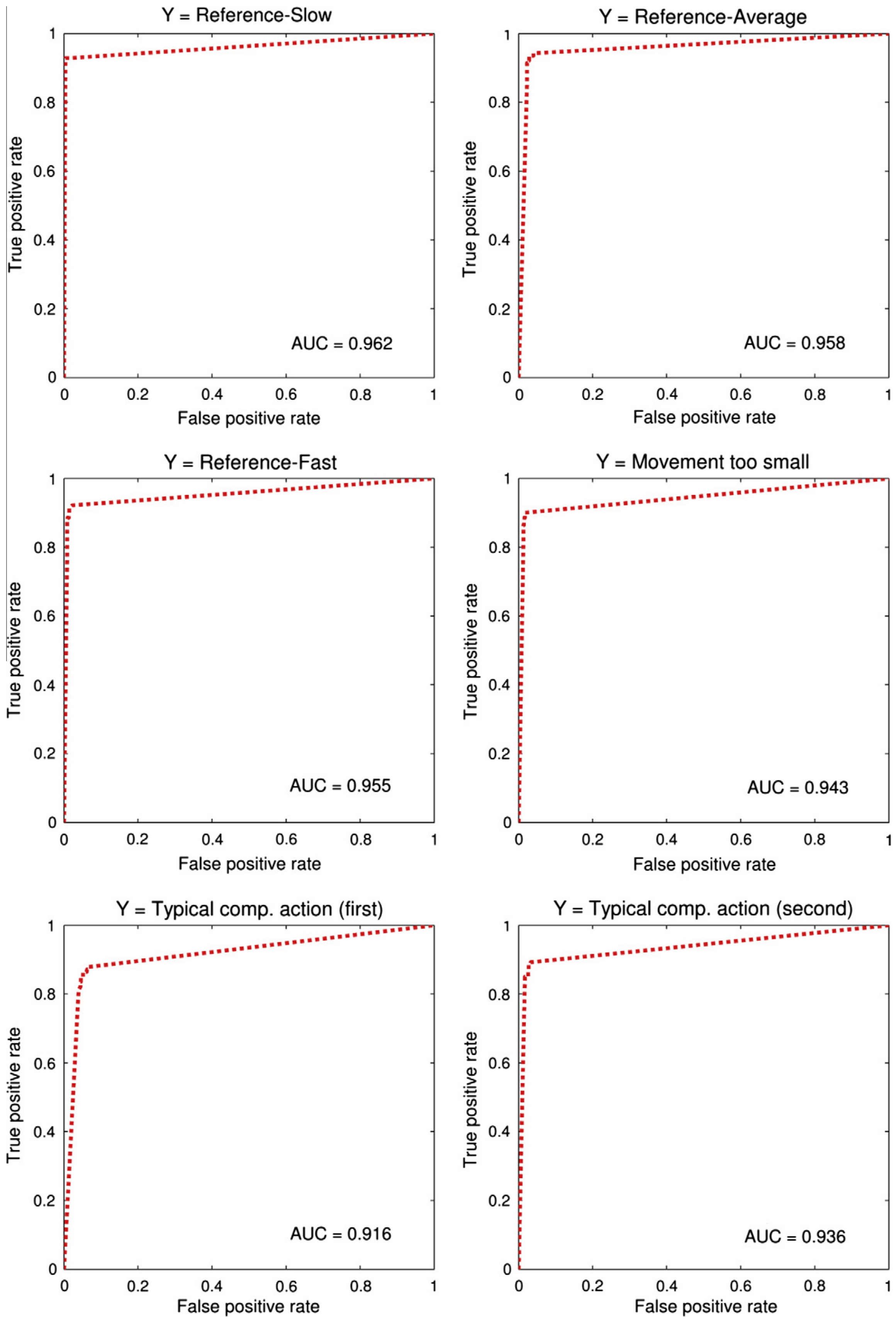

Fig. 9. ROC of the CTNB classifier for the *prediction of the execution mode* experiment in the case where the first half of each trajectory is used (6-class, half).

Fig. 10 concerns the *prediction of the execution correctness* experiment and shows the posterior probability of class $Y = Correct$, as computed by the CTNB classifier, for a sample trajectory which is known to be associated with an *Incorrect* movement. The CTNB model classifies correctly the sample trajectory as belonging to the class $Y = Incorrect$. The correct classification happens very early. Indeed, for times greater than 0.37 s the posterior probability of the class $Y = Correct$ is smaller than 0.04.

Fig. 11 concerns the *recognition of the execution correctness versus error type* experiment and shows the evolution along time of the posterior probability of the class, as computed by the CTNB classifier, for a sample trajectory which is known to be associated with a *Movement too small* movement. Also in this case the CTNB classifies the sample trajectory by the correct class, i.e. $Y = Movement\ too\ small$. It is worthwhile to mention that while from 0 to 0.77 s the most probable class is $Y = Reference$, for times greater

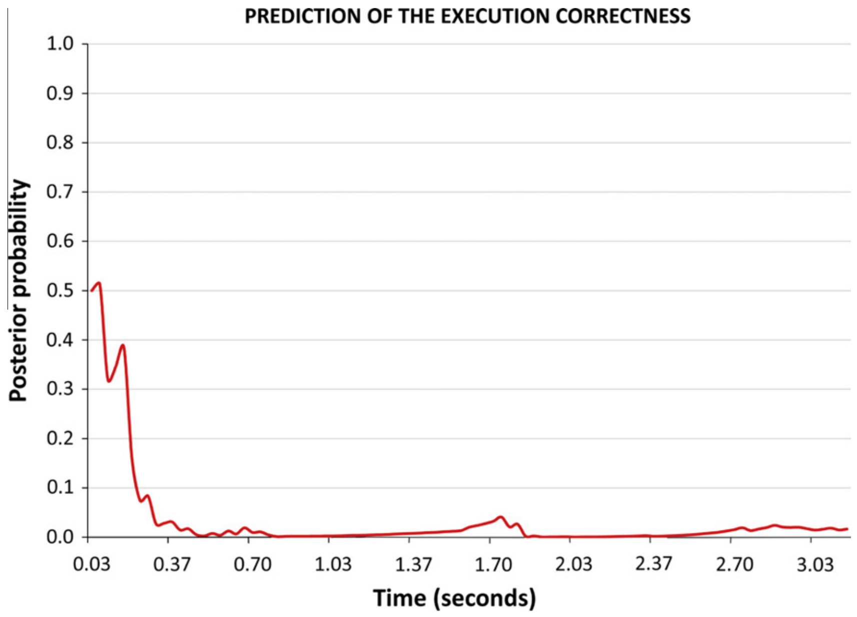

Fig. 10. *Prediction of the execution correctness*; evolution of the posterior probability of class $Y = Correct$, as computed by the continuous time naive Bayes classifier, for a sample trajectory which is known to be associated with an *Incorrect* movement.

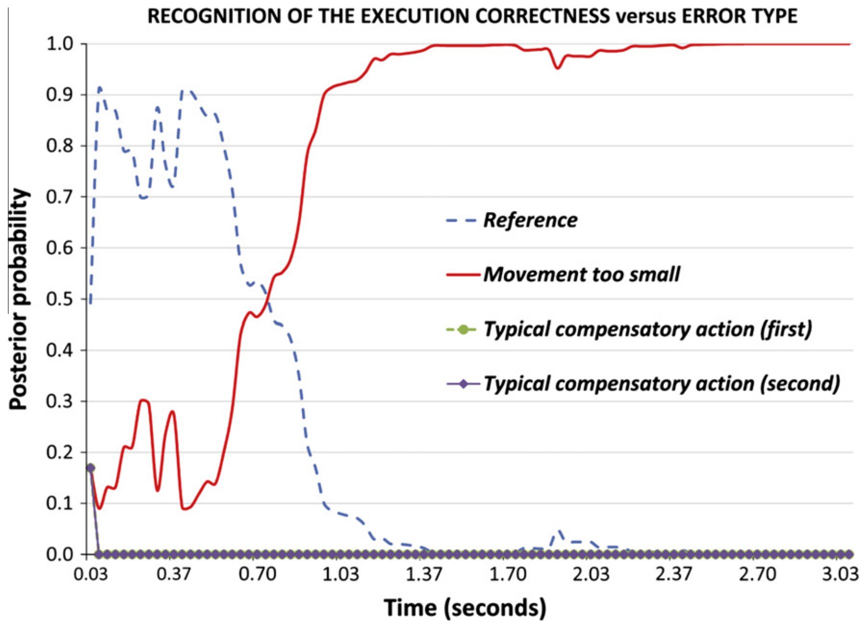

Fig. 11. *Recognition of the execution correctness versus error type*; evolution of the posterior probability of class $Y = Reference$, $Y = Movement\ too\ small$, $Y = Typical\ compensatory\ action\ (first)$ and $Y = Typical\ compensatory\ action\ (second)$, as computed by the continuous time naive Bayes classifier, for a sample trajectory which is known to be associated with a *Movement too small* movement. The line associated with the Typical compensatory action (second) overlaps the line associated with the Typical compensatory action (first).

Table 4
Average accuracy achieved by the CTNB classifier on the seven types of motor rehabilitation exercises (*Complete*).

i	2-class	4-class	6-class
1	0.967	0.975	0.883
2	1.000	0.992	0.950
3	0.967	0.958	0.950
4	1.000	0.992	0.942
5	0.992	0.933	0.917
6	0.992	0.992	0.917
7	1.000	0.983	0.950
Baseline	0.500	0.500	~0.167

than 0.77 s the most probable class is the true one, i.e. *Y* = *Movement too small*, whose posterior probability becomes greater than 0.9, even just after 0.9 s.

However, the comparison of the performance achieved by DTW, OE-DTW and CTNB, based on what is reported in Table 3 is on average accuracy values, i.e. accuracy values obtained by averaging accuracy values achieved for the seven types of motor rehabilitation exercise. Therefore, no conclusions can be drawn from the point of view of what the effectiveness of the CTNB classifier is with respect to the specific type of motor rehabilitation exercise. Therefore, it is useful to understand whether the average value of accuracy is significantly different with respect to the specific type of motor rehabilitation exercise. It may be the case that some types of motor rehabilitation exercises can be easily classified, while others not. In such a case it would be extremely useful to know which type/s of exercise/s is/are difficult to forecast. Therefore, the average value of the accuracy, achieved by the CTNB classifier, for each of the seven types of motor rehabilitation exercise is reported in Table 4.

5. Conclusions

In this paper the authors defined and studied a new class of probabilistic graphical models devoted to solving the problem of supervised classification in the case where: (i) the attributes are discrete, (ii) the time flows continuously, (iii) the attributes are fully observed and (iv) the class is expected to occur in the future.

Two instances from the class of continuous time Bayesian network classifiers, namely the continuous time naive Bayes and the continuous time tree augmented naive Bayes, have been defined. The learning algorithm for the continuous time naive Bayes classifier and the inference algorithm for classifying input trajectories by continuous time Bayesian network classifiers have been presented.

The class of continuous time Bayesian network classifiers, to the best of the authors' knowledge, is the first class of probabilistic graphical models devoted to solving the problem of supervised classification which directly models the time. Therefore, it is the only one allowing to recover the probability distribution over time when specific events occur. Models from this class are computationally efficient with respect to their static counterparts, which are difficult to apply in realistic contexts. The inference algorithm for continuous time Bayesian network classifiers allows us to make the classification decision at any point in time without needing to wait for the end of the trajectory. This is a particularly important feature for applicative domains as the one studied in this paper where the feedback to the patient must be given as soon as possible.

The CTNB classifier performs very well when compared with state-of-the-art algorithms, namely DTW and OE-DTW, and it significantly outperforms these algorithms when half trajectory is used. The CTNB classifier is particularly robust with respect to the complexity of the classification problem, i.e. number of classes. It achieves accuracy values which are more stable than those achieved by the DTW and OE-DTW algorithms.

A final remark is devoted to highlight that the proposed classifier is designed for discrete variables while motor rehabilitation data are continuous. Therefore, a discretization procedure has been developed. The performance of the classifier depends on the value of the parameters of the discretization procedure and a complex and time consuming tuning phase is required to achieve *optimal* results. Therefore, an improvement of the performances of the CTNB classifier, on the motor rehabilitation data, could possibly be achieved with additional tuning of the discretization procedure.

Acknowledgments

The authors would like to acknowledge the many helpful suggestions of the anonymous reviewers which helped to improve the paper clarity and quality with specific reference to numerical experiments and classification performance measures.

Appendix A

Algorithm 1. CTNB Learning

```
Require: a data set $\mathcal{D}$ of fully observed J-evidence-streams
    $\{(\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^{J_1}); ...; (\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^{J_{|\mathcal{D}|}})\}$, and the corresponding set of
    classes $\{y_1, y_2, ..., y_{|\mathcal{D}|}\}$, $y_j \in Val(Y)$, $j = 1, 2, ..., |\mathcal{D}|$.
Ensure: the CTNB classifier $\mathcal{C} = \{\aleph, P(Y)\}$ .
 1: for $k$=1 to $K$ do
 2:   P($k$) := 0;
 3: end for
 4: for $s$=1 to $|\mathcal{D}|$ do
 5:   P($y_s$) := P($y_s$) + $\frac{1}{|\mathcal{D}|}$;
 6: end for
 7: for $s$=1 to $|\mathcal{D}|$ do
 8:   $i$=1;
 9:   while $t_i \le T_s$ do
10:     for $n = 1$ to $N$ do
11:       $M(x_n^i, x_n^{i+1}, y_s) := M(x_n^i, x_n^{i+1}, y_s) + 1$;
12:       $T(x_n^i, y_s) := T(x_n^i, y_s) + (t_i - t_{i-1})$;
13:     end for
14:     $i$:=$i$+1;
15:   end while
16: end for
17: for $k = 1$ to $K$ do
18:   for $n = 1$ to $N$ do
19:     for $x \in Val(X_n)$ do
20:       $MM(x, y_k) := \sum_{x' \neq x} M(x, x', y_k)$;
21:       $q(x, y_k) := \frac{MM(x,y_k)}{T(x,y_k)}$;
22:       $\theta(x, y_k) := \frac{M(x,x',y_k)}{MM(x,y_k)}$;
23:     end for
24:   end for
25: end for
26: return $\aleph$, P.
```

Algorithm 1 learns the parameter of a CTNB classifier by exploiting an input data set $\mathcal{D}$ consisting of *fully observed J-evidence-streams*. It can be split in three parts; *lines 1–6* compute the prior probability for the class variable *Y* (returned by variable P), *lines 7–16* compute the sufficient statistics associated with each attribute X_n, $n = 1, 2, \ldots, N$, of the learning data set $\mathcal{D}$, i.e. $M(x,x',y_k)$ and $T(x,y_k)$ for each value y_k of the class variable *Y* while *lines 17–25* compute the corresponding MLE parameter estimates (which are returned embedded in the CTNB classifier $\aleph$).

Algorithm 2. CTBNC Inference

```
Require: a CTBNC $\mathcal{C} = \{\aleph, P(Y)\}$ consisting of $N$ attribute nodes and a class node $Y$ such that $Val(Y) = \{y_1, y_2, ..., y_K\}$, a fully observed J-evidence-stream $(\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^J)$ and a value for the parameter $\epsilon$.
Ensure: the maximum aposteriori classification $y^*$ for the fully observed J-evidence-stream $(\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^J)$.
1: for $k = 1$ to $K$ do
2:   $logp(y_k) \leftarrow \log P(y_k)$
3: end for
4: for $k = 1$ to $K$ do
5:   for $j = 1$ to $J$ do
6:     for $n = 1$ to $N$ do
7:       if $x_n^j \neq x_n^{j+1}$ then
8:         $logp(y_k) := logp(y_k) - q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})$
9:         $logp(y_k) := logp(y_k) + log\left(1 - exp\left(-q_{x_n^j x_n^{j+1}}^{pa(X_n)}\epsilon\right)\right)$
10:      else
11:        $logp(y_k) := logp(y_k) - q_{x_n^j}^{pa(X_n)}(t_j - t_{j-1})$
12:      end if
13:    end for
14:  end for
15: end for
16: $y^* \leftarrow \arg\max_{y \in Val(Y)} logp(y)$.
17: return $y^*$
```

Given $\mathcal{C} = \{\aleph, P(Y)\}$ consisting of *N* attribute nodes and a class node *Y* such that *Val*(*Y*) = {$y_1, y_2, \ldots, y_K$}, a *fully observed J-evidence-stream* ($\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^J$) and the value of the parameter ϵ, Algorithm 2 computes the posterior probability and returns the most probable class (11). The **for** statement in *line 1* computes the a priori probability for each class. The core of the algorithm, *lines from 4 to 15*, computes (11) by exploiting (7) (*lines* 8, 9 and 11). The **for** statement in *line* 4 ranges over the *K* classes, the **for** statement in *line* 5 ranges over the *J* evidence streams while the **for** statement in *line 6* ranges over the *N* CTBNC attributes. The most probable class y^* is computed by *line* 16.

References

[1] Rabiner LR. A tutorial on hidden Markov models and selected applications in speech recognition. Proc IEEE 1989;77(2):257–86.
[2] Dean T, Kanazawa K. A model for reasoning about persistence and causation. Comput Intell 1989;5(2):142–50.
[3] Dagum P, Galper A, Horvitz E. Dynamic network models for forecasting. In: Uncertainty in artificial intelligence; 1992. p. 41–8.
[4] Dagum P, Galper A. Forecasting sleep apnea with dynamic network models. In: Uncertainty in artificial intelligence; 1993. p. 64–71.
[5] Pavlovic V, Frey BJ, Huang TS. Time-series classification using mixed-state dynamic Bayesian networks. Comput Vis Pattern Recognit 1999;2:2609.
[6] Obermaier B, Guger C, Neuper C, Pfurtscheller G. Hidden Markov models for online classification of single trial eeg data. Pattern Recognit Lett 2001;22(12):1299–309.
[7] Andersson M. Classification of aerial missions using hidden Markov models. In: Nielsen T, Zhang N, editors. Symbolic and quantitative approaches to reasoning with uncertainty. Lecture notes in computer science. Berlin/Heidelberg: Springer; 2003. p. 125–36.
[8] Gu H, Ji Q. Facial event classification with task oriented dynamic Bayesian network. Comput Vis Pattern Recognit 2004;2:870–5.
[9] Riggelsen C, Ohrnberger M, Scherbaum F. Dynamic Bayesian networks for real-time classification of seismic signals. In: Kok J, Koronacki J, Lopez de Mantaras R, Matwin S, Mladenic D, Skowron A, editors. Knowledge discovery in databases: PKDD 2007. Lecture notes in computer science, vol. 4702. Berlin/Heidelberg: Springer; 2007. p. 565–72.
[10] Martínez M, Sucar LE. Learning dynamic naive Bayesian classifiers. In: FLAIRS conference; 2008. p. 655–9.
[11] Friedman N, Geiger D, Goldszmidt M. Bayesian network classifiers. Mach Learn 1997;29:131–63.
[12] Tucker A, Hoen PAC, Vinciotti V, Liu X. Temporal Bayesian classifiers for modelling muscular dystrophy expression data. Intell Data Anal 2006;10(5):441–55.
[13] Tucker A, Vinciotti V, Liu X, Garway-Heath D. A spatio-temporal Bayesian network classifier for understanding visual field deterioration. Artif Intell Med 2005;34(2):163–77.
[14] Fisher M, Gabbay D, Vila L. Handbook of temporal reasoning in artificial intelligence (foundations of artificial intelligence). New York, NY, USA: Elsevier Science Inc.; 2005.
[15] Combi C, Keravnou-Papailiou E, Shahar Y. Temporal information systems in medicine. 1st ed. Springer Publishing Company, Incorporated; 2010.
[16] Pearl J. Probabilistic reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann; 1988.
[17] Pearl J. Causality: models, reasoning and inference. Cambridge University Press; 2000.
[18] Nodelman U, Shelton C, Koller D. Continuous time Bayesian networks. In: Uncertainty in artificial intelligence; 2002. p. 78–387.
[19] 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. Artif Intell Med 2009;45(1):11–34.
[20] Jensen FV, Nielsen TD. Bayesian networks and decision graphs. 2nd ed. Springer-Verlag; 2007.
[21] Nodelman U, Horvitz E. Continuous time Bayesian networks for inferring users presence and activities with extensions for modeling and evaluation. Tech. Rep.; 2003.
[22] Boudali H, Dugan JB. A continuous-time Bayesian network reliability modeling, and analysis framework. IEEE Trans Reliab 2006;55(1):86–97.
[23] Xu J, Shelton CR. Continuous time Bayesian networks for host level network intrusion detection. In: ECML/PKDD (2); 2008. p. 613–27.
[24] Fan Y, Shelton CR. Learning continuous-time social network dynamics. In: Uncertainty in artificial intelligence; 2009. p. 161–8.
[25] Gatti E, Luciani D, Stella F. A continuous time Bayesian network model for cardiogenic heart failure. Flex Serv Manufact J 2011:1–20.
[26] Nodelman U, Koller D, Shelton CR. Expectation propagation for continuous time Bayesian networks. In: Uncertainty in artificial intelligence; 2005a. p. 431–40.
[27] Saria S, Nodelman U, Koller D. Reasoning at the right time granularity 2007:1–9.
[28] Fan Y, Xu J, Shelton CR. Importance sampling for continuous time Bayesian networks. J Mach Learn Res 2010;11:2115–40.
[29] El-Hay T, Friedman N, Kupferman R. Gibbs sampling in factorized continuous-time Markov processes. In: Uncertainty in artificial intelligence; 2008. p. 169–78.
[30] Nodelman U, Shelton C, Koller D. Expectation maximization and complex duration distributions for continuous time Bayesian networks. In: Uncertainty in artificial intelligence; 2005b. p. 421–30.
[31] Witten IH, Frank E. Data mining: practical machine learning tools and techniques with Java implementations. Morgan Kaufmann; 1999. ISBN: 1-55860-552-5.
[32] Bellazzi R, Zupan B. Predictive data mining in clinical medicine: current issues and guidelines. I J Med Inform 2008;77(2):81–97.
[33] Demsar J. Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 2006;7:1–30.