2020 16th IEEE International Conference on Automation Science and Engineering (CASE)
August 20-21, 2020, Online Zoom Meeting

An optimization-based process mining approach for explainable classification of timed event logs

Hugo De Oliveira[1,2], Vincent Augusto[1], Baptiste Jouaneton[2], Ludovic Lamarsalle[2], Martin Prodel[2] and Xiaolan Xie[1,3], *Fellow, IEEE*

***Abstract*— This paper addresses the problem of supervised classification of time event logs of two classes: positive and negative population. The key idea of this paper to explain classification is to determine some process model that fits well the positive event logs and poorly the negative ones. More specifically, we introduce formal definitions of event logs, process models and a replayability score that measures the fitness of a process model for a given event log. We then set the event log classification as an optimization problem for the determination of a process model that maximizes its replayability for the positive population and minimizes its replayability for the negative one. A tabu search algorithm is then proposed to solve this problem. The proposed algorithm is compared with three state-of-the-art classification algorithms on test cases of various complexity. It is shown to provide superior performances and a graphic representation of the process model of the positive event logs.**

I. Introduction

Data is a powerful resource. Different structures of data can be found, within a large spectrum of complexity. In the field of supervised learning, machine learning algorithms for classification have been widely used. The paradigm for state-of-the-art classification algorithms is matrix-shaped input data: each observation (row) is a vector of features (column). Once trained, classifier's predictions for new observations are based on feature similarity with training observations.

However, when data is structured in event logs, each observation is an ordered list of events and no longer a single vector of features. Distinctive characteristics (patterns) can be of different types, as for example a special event's occurrence, an event preceding another or a typical time between two events. Data engineering exists in order to transform an event log into a feature matrix ("flattening" process). This data preprocessing step is challenging because potential distinctive patterns of the event log data need to be kept for the classifier. Furthermore, it might lead to high dimension and sparse matrices, especially when considering time between events.

Even if predictive performance is the predominant criterion for model approval, human understanding is a key lever for acceptance and practical application of decisions. This is the case in healthcare, where the identification of patients with pathways being suspicious of developing future medical complications is valuable. It offers the opportunity to identify at-risk patients and to respond with personalized medical care and prevention. However, ensuring a high predictive performance is challenging, as patients' pathways are complex: high number of different medical events, variability of pathways' length, variability of time between events... The imbalance between groups of patients with and without a given complication is also a recurrent problem. Even if some preprocessing methods such as over/under-sampling exist in the literature, performances can be substantially impacted.

To tackle these scientific challenges, we propose in this paper an explainable method for classification of time event logs of two classes: positive and negative population. The main idea to predict and explain is to construct a process model that fits well the positive event logs and poorly the negative ones. The proposed framework which relies on *time grid process models* [1], has the following characteristics: (1) designed for event log data; (2) robust to imbalanced classes; (3) explainable through the obtained process model, which represents the knowledge extracted from the event log of the positive class.

This paper is organized as follows. Section II presents a brief literature review related to classification using event logs. Important definitions and notations are presented in Section III. In Section IV, the problem settings and the proposed methodology is introduced. Section V presents a design of experiments on simulated data to assess the proposed methodology performances. Finally, a conclusion and future perspectives are given in Section VI.

II. Literature review

Collecting real-life process data results in time-dependent event logs. Many fields are concerned such as healthcare, manufacturing industry, software engineering or telecommunication [2]. The use of unsupervised methods on event logs is helpful to extract knowledge from data. For that purpose, process mining has become state-of-the-art. The primary objective of Process Mining is to do process discovery, i.e. to represent a summarized model of the event log [3]. It has been used in healthcare to map care processes and clinical pathways [4]. For example, Prodel et al. [5] used linear integer programming to discover patients' pathways from hospital data. In 2018, authors presented a meta-heuristic to perform optimal discovery of clinical pathways [6]. An

Corresponding author: Hugo De Oliveira.
[1]H. De Oliveira, V. Augusto and X. Xie are with Mines Saint-Étienne, Univ Clermont Auvergne, CNRS, UMR 6158 LIMOS, Centre CIS, F - 42023 Saint-Étienne France (e-mails: hugo.de-oliveira@emse.fr; augusto@emse.fr; xie@emse.fr).
[2]H. De Oliveira, B. Jouaneton, L. Lamarsalle and M. Prodel are with HEVA, 186 avenue Thiers, F-69465, Lyon, France (e-mails: bjouaneton@hevaweb.com; llamarsalle@hevaweb.com; mprodel@hevaweb.com).
[3]X. Xie is also with the Antai College of Economics and Management, Shanghai Jiao Tong University, China.

enhancement has been proposed by De Oliveira et al. [1] to introduce time in optimal process models.

The use of process mining frameworks to perform predictions has been already presented in the literature. Examples are numerous, as for example the prediction of the time before the occurrence of an event in a process, or the probability of a given task to be performed [7]. Each of these tasks could be performed using different methods. In the case of next activity prediction, Ferilli et al. presented two methods based on the `WoMan` framework [8]. The implementation of a predictive model in each node of a process model to predict future steps taking into account patients' characteristics has also been proposed in [9].

Binary classification applied on event logs data has been addressed in the literature. If time is neglected and only a succession of events is analyzed, the classification of event log data problem is similar to sequence classification. For that, three types of studies can be identified [10]: (1) featured-based classification (extracting features from an event log to create a matrix input for a classifier model); (2) distance-based classification (using a similarity measure between sequences); (3) model-based classification. The two first types of approaches are used in bioinformatics for DNA (deoxyribonucleic acid) alignment, and based on models defined by [11], [12] in particular. Improvements of these methods are explored in the literature [13]. The third type gathers statistical models as Hidden Markov Models [14], [15].

For supervised prediction on event log data, a data preprocessing phase is generally needed: applying existing prediction algorithms on event logs is not straightforward, as the event log needs to be transformed into a feature matrix. This transformation is done automatically using features extraction or using experts' knowledge. Healthcare is a field of interest, as patients' pathways are defined by succession of medical events, time between events being a key indicator of care. State-of-the-art machine learning approaches have been widely used for prediction in healthcare. Case studies found in the literature are of various type [16], as prediction of diseases [17], mortality [18], prevention tests [19] and readmission [20], [21]. Medical features are generally selected by experts, but the longitudinal structure of patients' pathway is either lost during feature extraction or leads to a sparse representation of data [22]. Moreover, algorithms like Decision Tree or Logistic Regression are preferred by practitioners due to their explainability.

As a result, to the best of our knowledge, no classification algorithm has been designed for event log data, with a particular focus on learning explainability. This focus carries potential applications, such as healthcare and particularly patient's pathway constitutes the initial motivation of the development of such a methodology. The explainability of predictions for medical experts and decision makers is essential, especially when time is a possibly distinctive feature.

III. Preliminaries on event logs and process models

A. Event log

Definition 1: (Event). An event denoted e is defined as a couple (a, t) where $a \in A$ is an element of a finite set A of labels corresponding to the event class of e, and $t \in T$ with $T = \mathbb{N}$ or $\mathbb{R}$ is the event time or time-stamp. An event e is also defined by the labeling function $label(e) = a$ and the timing function $time(e) = t$.

Definition 2: (Trace). A trace is a sequence of events $\sigma = e_1, e_2, \ldots, e_m$ with $m \in \mathbb{N}^*$ such that $e_k \in A \times T$ and $time(e_k) < time(e_{k+1})$.

Definition 3: (Event log). An event log is a set of traces $L = \{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ with $n \in \mathbb{N}^*$. An event log contains all input data of this paper. It is assumed that each label appears at least once in the event log L, i.e. $\forall a \in A : \exists \sigma \in L, e \in \sigma \mid e = (a, t)$.

Definition 4: (Event diversity). The event diversity div_e is defined as $div_e = |A|$. This descriptor gives information about the variability of the event log in terms of labels.

B. Process models

Definition 5: (Time grid process model). A time grid process model of a given log L is a four-uplet $TG\text{-}PsM = (N, E, \mathcal{L}, \mathcal{T})$ where:

- N is a set of nodes partitioned into K disjoint subsets called layers, i.e. $N = N_1 \cup \cdots \cup N_k, N_k \cap N_l = \emptyset$;
- $E \subset N \times N$ is a set of edges such that $(x, y) \subset E$ with $x \in N_k, y \in N_l$ implies $k < l$, i.e. the process model is acyclic with edges going from lower layers to higher layers;
- $\mathcal{L} : N \to A$ is the labeling function of the nodes.
- $\mathcal{T} : E \to T \times T$ associates a time interval $[a_{(x,y)}, b_{(x,y)}]$ to each edge $(x, y) \in E$.

Interesting properties of such a process model are as follows. A same label can appear at different positions in the model. Constraints for edges link lower positions to strictly higher ones. This produces oriented process models, with no backward edge and possibly a same label found in lower an higher positions. Moreover, multiple edges can be found between two nodes, each edge having a different time characteristic. This time characteristic on edges serves to consider time during the optimization process.

In the following, all process models are supposed to be time grid process models, as defined in Definition 5.

C. Replayability

Definition 6: (Replayability). The replayability function is denoted $\mathcal{R}$, and returns the replayability score:

$$\mathcal{R}(\text{TG-PsM}, \sigma) \in [0, 1] \tag{1}$$

which is the representativeness of the trace σ by the process model $TG\text{-}PsM$. By extension, the replayability score distribution of an event log L is the set of replayability score values for each trace in L:

$$\mathcal{R}(\text{TG-PsM}, L) = (\mathcal{R}(\text{TG-PsM}, \sigma))_{\sigma \in L} \tag{2}$$

The replayability is used in [5], [6], [1] to evaluate the ability of a process model to represent a given trace. The result of the procedure is a replayability score between 0 and 1, where 1 corresponds to the best possible representation of a trace by a process model. The following elements are positively taken into account in the replayability: (1) nodes matching trace's events; (2) edges matching event transitions; (3) time characteristic of edges matching time-stamp of event logs; (4) no central event of the trace skipped. As the replayability score measures the ability of a process model to represent a given trace, the analysis of the replayability score distribution points out the representativeness of a process model regarding the entire event log. Further details about the replayability for time grid process models are given in [1].

IV. Process model-based classification of event logs data

The proposed approach is an optimization-based method, at the crossroad between machine learning, process mining and operational research. In this section, we formally define the problem settings and describe the methodology, with a particular focus on the optimization process involved.

A. Problem setting

The problem here consists of having two labeled event logs L_0 and L_1, and learn patterns from this data in order to predict for new, unlabeled traces. In other words, the problem addressed here is a binary classification problem, with data involved being event log of traces (and not sets of labeled vectors described by features as in classic binary classification).

Lets define a binary classed event log:

$$L^{train} = (L_0^{train}, L_1^{train}) \tag{3}$$

where traces from L_k^{train} are of class k for $k \in \{0, 1\}$. For a given process model $TG\text{-}PsM$, let

$$\mathcal{R}_0^{train} = \mathcal{R}(\text{TG-PsM}, L_0^{train}) \tag{4}$$

and

$$\mathcal{R}_1^{train} = \mathcal{R}(\text{TG-PsM}, L_1^{train}) \tag{5}$$

be the replayabilities of traces of L_0^{train} and L_1^{train}, respectively. Resulting replayability distributions can be visualized on a single plot, as shown in Figure 1. Supposing that the $TG\text{-}PsM$ better represents traces from the positive class than traces from the negative one, replayability scores from $\mathcal{R}_1^{train}$ will be generally higher than replayability scores from $\mathcal{R}_0^{train}$. The process of training a $TG\text{-}PsM$ consists in creating such a process model.

B. Process model-based classification

The process model-based classification algorithm is composed of 2 steps:

1) **Train**: construct a $TG\text{-}PsM$ from L^{train} to get replayability distributions $\mathcal{R}_0^{train}$ and $\mathcal{R}_1^{train}$;

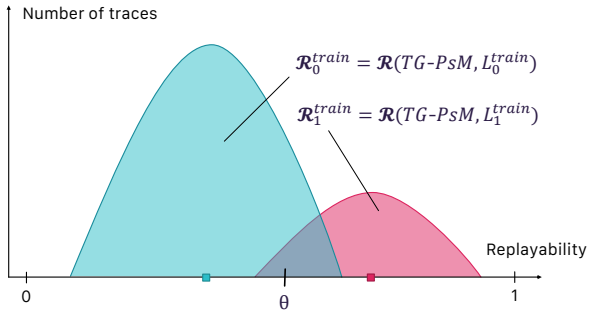

Fig. 1: Replayability graph of a process model $TG\text{-}PsM$ on training data $(L_0^{train}, L_1^{train})$ with threshold θ for classification.

2) **Predict**: for a new trace σ, compute its replayability $\mathcal{R}(TG\text{-}PsM, \sigma)$ and return the corresponding predicted class by comparing it to a given threshold θ.

The choice of the threshold θ is a widespread issue for binary classifiers, to predict in practice for every individuals. To find the best split between the two replayability distributions, the threshold which minimizes the gini impurity is chosen. One can infer that the construction of the process model (the training of $TG\text{-}PsM$) is the key to improve classification performances. The main idea here is to build a process model which produces distinct distributions for both training classes on the replayability graph. A Tabu search is used, motivated by previous work [6], [1]. Before detailing the search algorithm, two objective functions are described.

C. Objective functions for process model optimization

Two objective functions are presented, each involving a different measure of process model quality. The average replayability function is denoted as:

$$\overline{\mathcal{R}}(\text{TG-PsM}, L) = \frac{1}{|L|} \sum_{\sigma} \mathcal{R}(\text{TG-PsM}, \sigma) \tag{6}$$

1) RepOpt: The first objective function consists in searching a final solution which maximizes the mean replayability of the event log L_1^{train} (positive class):

$$\overline{\mathcal{R}}(\text{TG-PsM}, L_1^{train}) \qquad \text{(RepOpt)}$$

One can notice that elements of L_0^{train} stay unused during the optimization process. This objective function was used in process discovery for unsupervised process mining [1].

2) DiffOpt: Instead of maximizing the replayability of the traces of the positive class, we maximize the difference between the means of the two classes. The idea is to construct a graph that best replays traces of L_1^{train} and that replays badly traces of L_0^{train}.

$$\overline{\mathcal{R}}(\text{TG-PsM}, L_1^{train}) - \overline{\mathcal{R}}(\text{TG-PsM}, L_0^{train}) \qquad \text{(DiffOpt)}$$

Expectations with this objective function is the evacuation of shared patterns between positive and negative classes, while keeping those specific to the positive one.

The two previously defined objective functions (RepOpt) and (DiffOpt) constitute the core of the local search procedure used to create optimal process models. This procedure is detailed thereafter.

D. Tabu search for process model optimization

The proposed methodology to fit a process model $TG\text{-}PsM$ on train data L^{train} is an optimization process based on a local search. Starting from a random solution (i.e. a process model), a neighborhood of solutions is created. Each neighbor is a slightly modified version of the current process model (2 possible moves: add a new promising node or randomly delete a node). Each neighbor is then evaluated by computing the objective function (RepOpt) or (DiffOpt). The best neighbor is kept as the current solution and added to a fixed sized first-in-first-out list of tabu solutions. Tabu solutions cannot be selected when creating a neighborhood. This process is iterated until a stopping criterion is reached (a total maximum number of iterations or a maximum number of iterations without any improvement). Resulting process model is the best evaluated solution encountered during the entire search.

Required parameters for the optimization procedure are the constraints (the maximum number of nodes U_N, the maximum number of edges U_E and the maximum position in the process model p_{max}) and search parameters (the neighborhood size, the tabu list size, and the stopping criteria). The set of time intervals for edges is also an input parameter. Pertinent time intervals are constructed using Kernel Density Estimation, previously proposed in [1].

V. Numerical experiment

The classification methodology is validated through the following design of experiments. Event logs $(L_0, L_1) = (L_0^{train}, L_0^{test}, L_1^{train}, L_1^{test})$ are build with different hidden patterns, the objective being to learn from training event logs and accurately predict for test ones.

A. Data generation

Two graphs G_0 and G_1 are constructed, constituted of nodes arranged in layers having a maximum number of identical positions equal to p_m. For each position $p \in [\![1, p_m]\!]$, the corresponding layer is composed of n = div_e different nodes. Then, a proportion of shared patterns is removed, by deleting $c_{pat} * |N|$ randomly chosen nodes from G_1 and corresponding edges. An illustration for G_0 and G_1 is shown on the left of Figure 2.

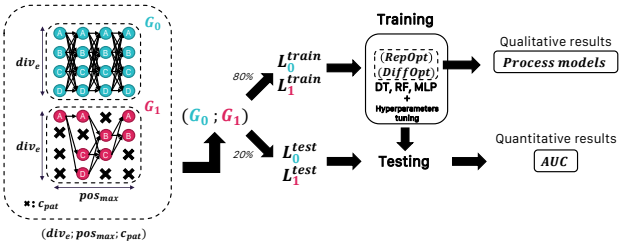

Fig. 2: Schematic representation of the design of experiments.

A trace σ is created by starting from the lower position of the graph, and by adding to the trace an event with the same label as a randomly selected node of the next increasing position. The time-stamp of the new event is added to the previous time-stamp; it is a time value randomly selected for L_0 and L_1 following respectively $\mathcal{U}(0, 400)$ and $\mathcal{N}(200, 25)$. At each step the process can be stopped with a probability $p = \frac{\mathcal{P}(n_{current})}{p_m}$, where $n_{current}$ is the current node of G, corresponding to the last addition to σ. Such event log construction process ensures the presence of a pattern in G_1, in terms of labels, transitions and time. The probability of stopping the construction process ensures a variability in traces' lengths. The higher c_{pat} is, the smaller G_1 and the more specific the process model will be. Event logs dimensions are noted $N = |L_0^{train}|$ and $P = |L_1^{train}|$. The design of experiments consists in testing different configurations for div_e, p_m and c_{pat}. A summary of parameters for the design of experiment is presented in Table I.

TABLE I: Search parameters and constraints used for design of experiments.

Data parameters	**Value**
Number of traces	$N = 1800$ and $P = 200$
Diversity of events	$div_e \in [10, 50, 100]$
Maximum length of generated traces	$p_m \in [10, 25, 50]$
Event pattern coefficient	$c_{pat} \in [0.90, 0.75]$
Time transition patterns	G_0: $\mathcal{U}(0, 400)$
	G_1: $\mathcal{N}(200, 25)$
Graph parameters	**Value**
Maximum number of nodes	$U_N = 0.2 \times div_e \times p_{max}$
Maximum number of edges	$U_E = 2 \times U_N$
Maximum number of positions	$p_{max} = \lvert\sigma\rvert_{max}$
Tabu search parameters	**Value**
Neighborhood size	15
Size of Tabu list	10
Max. number of iterations	500
Max. number of iterations without improvement	15

B. Evaluation metrics

ROC (Receiver Operating Characteristic) curve is the true positive rate (tpr) in function of false positive rate (fpr). This curve is obtained by varying the threshold for prediction (θ for process model classifier). The AUC (Area Under the Curve) is chosen as the performance measure, justified by the presence of imbalanced classes.

C. Benchmark of binary classifiers

The process model-based classifier is compared with three state-of-the-art machine learning algorithms for binary classification: Decision Tree (DT), Random Forest (RF) and feed-forward Multi-layer Perceptron (MLP). These methods expect matrix-shaped input data, so a "flattening" preprocessing is applied to the event log: features are created by combining every possible event's labels with corresponding time-stamps encountered in the event log. For each trace having a given event at a given time stamp, the corresponding

feature value is set to 1, 0 otherwise. The advantage of this preprocessing approach is to provide the 3 machine learning models with the most accurate data possible. The inconvenience is the high dimension and sparsity of input data. Because imbalanced classes are an issue for binary classification algorithms, data balancing has been applied before fitting DT, RF and MLP. An oversampling of the minority class was applied using the SMOTE algorithm [23]. Moreover, a high-dimension grid of hyperparameters was defined, and a random search on it was performed. A 3-fold cross-validation was used on the training set to determine the best hyperparameter combination for each data set. The previously described design of experiment is summarized in Figure 2. Calculations were done in `python 3.6`, using `scikit-learn` library for DT, RF and MLP methods.

D. Results

1) Quantitative results: For each descriptor combination, median and standard deviation of AUC on test sets for 10 replications are presented in Table II. The best average AUC score is highlighted in bold.

Our method with objective function (DiffOpt) globally outperforms DT, RF and MLP in most settings. The gap between proposed methods and state-of-the-art algorithms increases with the increase of div_e for $c_{pat} = 0.90$. When patterns in event logs of the positive class are less specific ($c_{pat} = 0.75$), the general performances decreases and variability increases. (DiffOpt) seems robust regarding the increase in diversity of events (div_e) and the increase in traces' size (p_m). Other methods are negatively impacted by the increase in diversity and trace size which results in reduced AUC performances.

TABLE II: Benchmark of AUC for 5 methods: average and standard deviation.

Data			DT		RF		MLP		(RepOpt)		(DiffOpt)	
c_{pat}	div_e	p_m	AVG	STD	AVG	STD	AVG	STD	AVG	STD	AVG	STD
0.90	10	10	0.96	0.01	0.96	0.01	0.99	0.01	0.99	0.01	**1.00**	0.00
		25	0.95	0.01	0.95	0.01	**1.00**	0.00	0.99	0.02	0.99	0.02
		50	0.96	0.01	0.96	0.01	**1.00**	0.00	0.99	0.02	0.98	0.01
	50	10	0.95	0.02	0.95	0.02	0.97	0.02	0.98	0.01	**1.00**	0.00
		25	0.95	0.02	0.95	0.02	0.97	0.01	0.97	0.01	**0.99**	0.00
		50	0.95	0.03	0.95	0.03	0.98	0.02	0.97	0.01	**0.99**	0.00
	100	10	0.95	0.01	0.95	0.01	0.96	0.01	0.98	0.01	**0.99**	0.00
		25	0.92	0.05	0.92	0.05	0.97	0.01	0.98	0.02	**0.99**	0.00
		50	0.90	0.07	0.90	0.07	0.97	0.02	0.97	0.01	**0.99**	0.00
0.75	10	10	0.88	0.05	0.88	0.05	0.94	0.03	0.95	0.05	**0.97**	0.02
		25	0.89	0.04	0.90	0.04	0.95	0.04	0.95	0.06	**0.96**	0.02
		50	0.85	0.06	0.86	0.06	0.93	0.04	**0.94**	0.04	0.91	0.06
	50	10	0.88	0.03	0.88	0.03	0.86	0.06	0.90	0.03	**0.95**	0.02
		25	0.87	0.04	0.85	0.06	0.87	0.05	0.91	0.04	**0.94**	0.01
		50	0.88	0.03	0.86	0.06	0.85	0.08	0.87	0.03	**0.94**	0.02
	100	10	0.77	0.10	0.78	0.06	0.86	0.03	0.87	0.05	**0.93**	0.02
		25	0.65	0.06	0.64	0.07	0.80	0.11	0.81	0.04	**0.92**	0.02
		50	0.64	0.07	0.63	0.06	0.84	0.05	0.72	0.05	**0.86**	0.07

2) Qualitative and explainable results: The interpretability is a crucial motivation in this study. Two examples of process models obtained after training (using (DiffOpt) objective function) are displayed in Figures 3 and 5. Process models are read from left to right, following increasing node positions. Circles represent nodes of the model, and flux from circles represent edges. Node size and edge size are proportional to the number of traces from L_1^{train} replayed during the replayability game. Each obtained process model graphically highlights distinctive patterns, mined during the training optimization. Thus, for simulated event log with high pattern coefficient ($c_{pat} = 0.9$) and narrow dimensions ($div_e = 10$ and $p_{max} = 10$), the resulting process model is simple (Figure 3). However, its power to distinct traces is strong, as highlighted by AUC performances (AUC $= 1.00 \pm 0.00$).

Fig. 3: Example of process model obtained using (DiffOpt), with $c_{pat} = 0.9$, $div_e = 10$ and $p_{max} = 10$.

To illustrate the prediction method, an example is presented in the following.

Example 1: An event log containing 2 traces σ_A and σ_B is presented in Figure 4. We want to predict if these traces are of class 0 or 1, according to the process model of Figure 3. After training on L_1^{train} and L_0^{train}, the process model $TG\text{-}PsM$ maximizes (DiffOpt). Thus, the mean replayability of traces of L_1^{train} (0.98) is much higher than the mean replayability of traces of L_0^{train} (0.32). The threshold minimizing the gini impurity on the two training replayability distributions is $\theta = 0.40$. By computing the replayabilities of both traces, it appears that σ_A is well replayed (0.80), while σ_B replayability is pretty low (0.25). After a comparison to the threshold θ, class 1 and class 0 are attributed to σ_A and σ_B, respectively.

id	time-stamp	event
A	0	label_5
A	12	label_4
A	25	label_0
A	28	label_5
A	31	label_8
B	0	label_8
B	15	label_9
B	42	label_0
B	51	label_4

$$\bar{\mathcal{R}}(TG\text{-}PsM, L_0^{train}) = 0.32$$
$$\bar{\mathcal{R}}(TG\text{-}PsM, L_1^{train}) = 0.98$$

Predictions ($\theta = 0.40$)

$$\mathcal{R}(TG\text{-}PsM, \sigma_A) = 0.80 > \theta \rightarrow 1$$
$$\mathcal{R}(TG\text{-}PsM, \sigma_B) = 0.25 < \theta \rightarrow 0$$

Fig. 4: Event log of σ_A and σ_B (left) and predictions (right).

A more complex pattern extraction is presented in Figure 5, with $c_{pat} = 0.75$, $div_e = 10$ and $p_{max} = 50$. The process model is characterized by two central events ("*label_6*" and "*label_9*"), surrounded by other spread out and less specific ones. As the process model definition carry time characteristics on edges, potential distinctive time patterns are also extracted. Time-transitions for the class 1 followed $\mathcal{N}(200, 25)$ (and $\mathcal{U}(0, 400)$ for class 0). Thus, examples of time interval retained by the model (for example $[88, 264]$ and $[96, 289]$ in Figures 5), validate the ability of the method to display hidden time patterns.

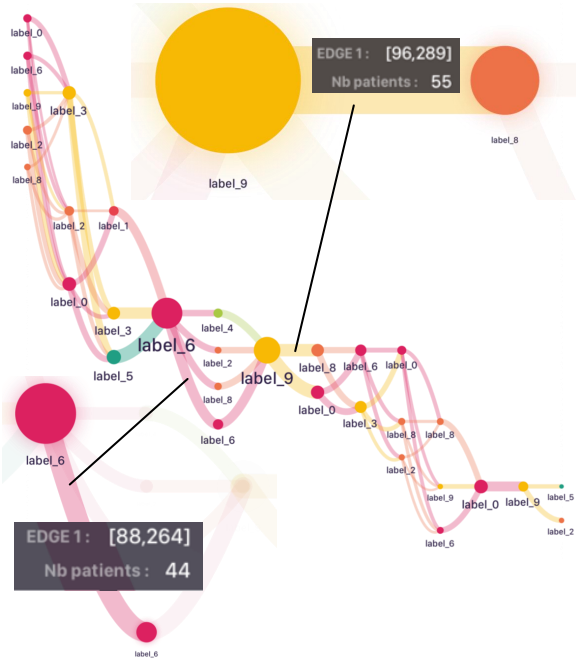

Fig. 5: Example of process model obtained using (DiffOpt), with $c_{pat} = 0.75$, $div_e = 10$ and $p_{max} = 50$.

VI. Conclusion

In this article, a new method for binary classification on timed event logs is proposed. Numerical experiments on synthetic data are presented, the robustness of the method being tested on event logs of increasing complexity. Quantitative results demonstrate the ability of the (DiffOpt) method to give outstanding performances in terms of AUC. Comparisons with state-of-the-art machine learning methods show the competitiveness of the proposed binary classifier when directly applied on imbalanced event logs with no use of over/under-sampling on training data. As process models carry distinctive patterns discovered during the training process, displaying them graphically illustrate future predictions.

Limitations and opportunities for future work are the following. Multi-class classification is not directly treated in this paper, but one can switch from binary to multi-class classification through "one-versus-all" settings. The current model cannot be updated with new traces batch. It must be entirely rebuilt. However, starting a new optimization process with already trained model as the initial solution could be a good strategy. The simulated event logs used here were designed to mimic patterns which will be interesting to found in clinical pathways extracted from claim databases. As the presented methodology is promising on synthetic data, future work will focus on practical healthcare case studies.

References

[1] H. De Oliveira, V. Augusto, B. Jouaneton, L. Lamarsalle, M. Prodel, and X. Xie, "Optimal process mining of timed event logs," *Information Sciences*, vol. 528, pp. 58 – 78, 2020.

[2] A. R. C. Maita, L. C. Martins, C. R. L. Paz, L. Rafferty, P. C. K. Hung, S. M. Peres, and M. Fantinato, "A systematic mapping study of process mining," *Enterprise Information Systems*, vol. 12, no. 5, pp. 505–549, 2018.

[3] W. M. P. van der Aalst, "Introduction," in *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, pp. 1–25, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011.

[4] T. G. Erdogan and T. Ayca, "Systematic mapping of process mining studies in healthcare," *IEEE Access*, vol. 6, pp. 1–1, 04 2018.

[5] M. Prodel, V. Augusto, X. Xie, B. Jouaneto, and L. Lamarsalle, "Discovery of patient pathways from a national hospital database using process mining and integer linear programming," in *CASE*, pp. 1409–1414, 2015.

[6] M. Prodel, V. Augusto, B. Jouaneton, L. Lamarsalle, and X. Xie, "Optimal process mining for large and complex event logs," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1309–1325, 2018.

[7] W. M. P. van der Aalst, *Operational Support*, pp. 301–321. Berlin, Heidelberg: Springer Berlin Heidelberg, 2016.

[8] S. Ferilli and S. Angelastro, "Activity prediction in process mining using the WoMan framework," *Journal of Intelligent Information Systems*, vol. 53, pp. 93–112, 2019.

[9] M. Prodel, V. Augusto, X. Xie, B. Jouaneton, and L. Lamarsalle, "Stochastic simulation of clinical pathways from raw health databases," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pp. 580–585, Aug 2017.

[10] Z. Z. Xing, J. Pei, and J. K. Eamonn, "A brief survey on sequence classification," *SIGKDD Explorations*, vol. 12, pp. 40–48, 11 2010.

[11] S. B. Needleman and C. D. Wunsch, "A general method applicable to the search for similarities in the amino acid sequence of two proteins," *Journal of Molecular Biology*, vol. 48, no. 3, pp. 443 – 453, 1970.

[12] T. Smith and M. Waterman, "Identification of common molecular subsequences," *Journal of Molecular Biology*, vol. 147, no. 1, pp. 195 – 197, 1981.

[13] B. Chowdhury and G. Garai, "A review on multiple sequence alignment from the perspective of genetic algorithm," *Genomics*, vol. 109, no. 5, pp. 419 – 431, 2017.

[14] Y. Byung-Jun, "Hidden markov models and their applications in biological sequence analysis," *Current Genomics*, vol. 10, no. 6, pp. 402–415, 2009.

[15] S. Blasiak and H. Rangwala, "A hidden markov model variant for sequence classification.," *IJCAI International Joint Conference on Artificial Intelligence*, pp. 1192–1197, 01 2011.

[16] H. De Oliveira, M. Prodel, and V. Augusto, "Binary classification on french hospital data: Benchmark of 7 machine learning algorithms," in *2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1743–1748, 10 2018.

[17] M. Nilashi, O. Ibrahim, H. Ahmadi, and L. Shahmoradi, "An analytical method for diseases prediction using machine learning techniques," *Computers & Chemical Engineering*, vol. 106, 06 2017.

[18] A. Salcedo-Bernal, M. Villamil-Giraldo, and A. Moreno-Barbosa, "Clinical data analysis: An opportunity to compare machine learning methods," *Procedia Computer Science*, vol. 100, pp. 731 – 738, 2016. International Conference on Health and Social Care Information Systems and Technologies 2016.

[19] N. Herazo-Padilla, V. Augusto, B. Dalmas, X. Xie, and B. Bongue, "Screening a portfolio of pathologies by subject profiling and medical test rationing," in *2019 15th IEEE Conference on Automation Science and Engineering (CASE)*, pp. 424–430, Aug 2019.

[20] O. Ben-Assuli, R. Padman, M. Leshno, and I. Shabtai, "Analyzing hospital readmissions using creatinine results for patients with many visits," *Procedia Computer Science*, vol. 98, pp. 357 – 361, 2016. The 7th International Conference on Emerging Ubiquitous Systems and Pervasive Networks (EUSPN 2016).

[21] D. Hooijenga, R. Phan, V. Augusto, X. Xie, and A. Redjaline, "Discriminant analysis and feature selection for emergency department readmission prediction," in *IEEE Symposium Series on Computational Intelligence, SSCI 2018, Bangalore, India, November 18-21, 2018*, pp. 836–842, 2018.

[22] M. Vandromme, J. Jacques, J. Taillard, A. Hansske, L. Jourdan, and C. Dhaenens, "Extraction and optimization of classification rules for temporal sequences: Application to hospital data," *Knowledge-Based Systems*, vol. 122, pp. 148 – 158, 2017.

[23] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *Journal of Artificial Intelligence Research*, vol. 16, p. 321–357, Jun 2002.