MEDINFO 2023 — The Future Is Accessible
J. Bichel-Findlay et al. (Eds.)

doi:10.3233/SHTI231080

Automated Process Mining and Learning of Therapeutic Actions in the Intensive Care Unit

Anna ROMANOV[a,1] and Yuval SHAHAR[b]
[a] *Department of Software and Information Systems Engineering, Israel*
[b] *Ben-Gurion University of the Negev, Beer Sheva, Israel*
ORCiD ID: Yuval Shahar https://orcid.org/0000-0003-0328-2333

Abstract. In this study, we implemented a hybrid approach, incorporating temporal data mining, machine learning, and process mining for modeling and predicting the course of treatment of Intensive Care Unit (ICU) patients. We used process mining algorithms to construct models of management of ICU patients. Then, we extracted the decision points from the mined models and used temporal data mining of the periods preceding the decision points to create temporal-pattern features. We trained classifiers to predict the next actions expected for each point. The methodology was evaluated on medical ICU data from the hypokalemia and hypoglycemia domains. The study's contributions include the representation of medical treatment trajectories of ICU patients using process models, and the integration of Temporal Data Mining and Machine Learning with Process Mining, to predict the next therapeutic actions in the ICU.

Keywords. Process mining, temporal data mining

1. Introduction

The Intensive Care Unit (ICU) treats a variety of difficult and complex patients who are acutely unwell and require critical medical care. There have been several attempts to predict a clinician's next action, using data-science methods, especially in the ICU context [1]. However, none of them exploited the advantages inherent in Business Process Mining methods, in integration with time-oriented Data Science methods.

In this study, our objective is to propose a method for predicting the next course of action in the treatment of ICU patients using temporal data mining integrated with process mining. We developed an explainable framework that can model existing clinical care trajectories in the ICU for multiple patients, and, given the complete data, up to a given point in time, of a specific patient, predict the next actions that would usually be performed in one or more preceding time windows, each of several hours, in the future. The predictions can support the decisions of less experienced attending clinicians.

The methodology was evaluated on medical data from the hypokalemia and hypoglycemia domains using the MIMIC-IV data [2].

[1] Corresponding Author: Anna Romanov, email: annarom99@gmail.com

2. Method

Our methodology operates within bounded time windows following the onset of a medical condition of interest, denoted by t_0. The patient's trajectory up to t_0 is referred to as the *prefix*. The *suffix*, accordingly, includes all actions performed by physicians as part of the treatment, following t_0 and up to the end of the relevant time window. Figure 1 presents the architecture of the methodology we had designed, implemented, and evaluated, consisting of several steps, through which training sets are constructed and trained with the objective of predicting the next actions. The output is a process model that can be traversed to determine the most suitable treatment for a given patient.

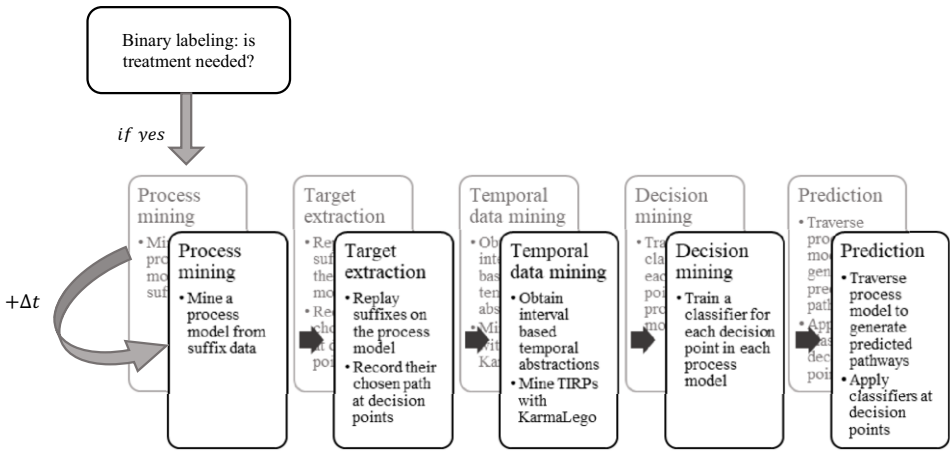

Figure 1. The suggested architecture, consisting of six consecutive steps that can be repeated for multiple time windows. TIRPs = Time-Interval Relations Patterns. KarmaLego = a computational module that discovers frequent TIRPs in multivariate, time-oriented data.

2.1. Binary Labeling

First, we preprocess the data and identify t_0 points. Data are labeled into two classes: the *positive* class, consisting of patients who were administered therapy within the time window of interest (following t_0), and the *negative* class, consisting of patients who did *not* get treatment. This labeling is used to train a classifier to determine whether treatment is needed. Only data from the positive class, in which *some* treatment was administered, are used for the construction of the process model in the next step.

2.2. Process Mining

In the second stage, a process model is obtained from *suffix* data of patients who got treatment (Figure 2). The process model represents the typical trajectories found in the data, including decision points. For this purpose, two popular process mining algorithms were tested: *heuristic miner* [3] and *IMf* [4].

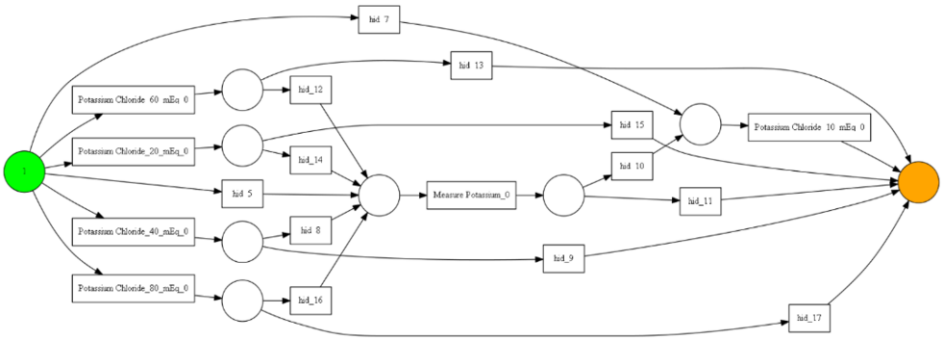

Figure 2. Process model example: hypokalemia (first time window). Squares represent possible actions; Circles help to define relations between actions; Arcs define the flow of trajectories in the model.

2.3. Target Extraction

It is important to realize *that each and every decision point in the process model defines a classification problem*, with outgoing arcs representing the different classes of possible treatments (Figure 2). Patients whose treatment trajectory passed through the decision point are added to its dataset, with the chosen arc as the label.

2.4. Feature Extraction

Abstracted time intervals [5] are discovered from the *prefix* data and mined for *time-interval relations patterns* (TIRPs) using the KarmaLego TIRP-discovery algorithm [6], whose main parameter is the minimal TIRP vertical support (VS), i.e., percentage of patients in which the TIRP was detected at least once.

Frequent TIRPs are discovered for each patient and are aggregated to be used as features, as originally suggested by Moskovitch and Shahar [7]: either as a binary indicator, indicating whether the TIRP has been observed for the patient or not; or level of *horizontal support* (HS), indicating the number of times that each TIRP was observed for the given patient; or *mean duration* (MeanD): the mean duration, from start to end, of each TIRP discovered for the given patient. The TIRP features are used to train the decision-point classifiers.

2.5. Decision-Point-Based Training and Classification

A training set was constructed, and a classifier was trained, *for each decision point* in the model discovered by process mining. XGBoost [8] was our induction algorithm.

2.6. Prediction

Given the trained model and a patient at time t_0, with a key abstraction such as Hypokalemia detected, the patient's data up to t_0 is processed and mined for temporal intervals and TIRPs, from which, in turn, features such as HS(TIRP) or MeanD(TIRP) are extracted. Only TIRPs encountered during training are used. The binary classifier is applied to the features, to determine whether treatment is needed. If the prediction is positive, the trained process model is traversed to determine the next action: iteratively,

whenever a decision point is reached, the corresponding classifier (trained for that decision point) is applied to determine the most likely continuation.

3. Results

3.1. Process Mining

First, we examined how well the process mining works on the ICU data, using standard process-mining measures. In all time windows and domains, results have reached a *fitness* of above 0.9 (the model represents well the behavior seen in the data) and a *precision* of above 0.95 (the model mostly does not allow behavior not seen in the data), with the best performance reached by the heuristic miner.

3.2. Classification

Tables 1 and 2 present the classification results for the first four and eight hours after encountering a Hypoglycemia or Hypokalemia state, respectively. Table 3 presents the Jaccard index of the predicted sequences for both domains (See Discussion for details).

Table 1. Classification results in the decision points of the first time window data of Hypoglycemia.

Decision point	VS threshold	Feature type	AUC	Recall	Specificity
Dextrose_10_gr_0	0.4	MeanD	0.671	0.433	0.859
Dextrose_25_gr_0	0.4	Binary	0.715	0.655	0.704
Dextrose_50_gr_0	0.2	Binary	0.612	0.601	0.662
Dextrose_5_gr_0	0.4	HS	0.769	0.659	0.815
Dextrose_above 50_gr_0	0.1	HS	0.715	0.729	0.609
Measure Glucose_0	0.3	HS	0.546	0.454	0.780
source0	0.1	MeanD	0.656	0.392	0.795
MeasureGlucose1MeasureGlucose0	0.1	Binary	0.663	0.454	0.780
binary	0.1	MeanD	0.972	0.907	0.924

Table 2. Classification results in the decision points of the first time window data of Hypokalemia.

Decision point	VS threshold	Feature type	AUC	Recall	Specificity
Measure Potassium_0	0.4	MeanD	0.691	0.857	0.417
Potassium Chloride_10_mEq_0	0.4	MeanD	0.710	0.798	0.667
Potassium Chloride_20_mEq_0	0.1	MeanD	0.665	0.779	0.410
Potassium Chloride_40_mEq_0	0.1	HS	0.641	0.563	0.664
Potassium Chloride_60_mEq_0	0.1	Binary	0.639	0.487	0.766
Potassium Chloride_80_mEq_0	0.1	Binary	0.663	0.487	0.766
source0	0.1	Binary	0.537	0.234	0.799
binary	0.1	Binary	0.715	0.600	0.708

Table 3. Jaccard index of the predicted sequences, by domain and time window.

Domain	Time window	Jaccard index
Hypokalemia	first	0.363
	second	0.283
Hypoglycemia	first	0.370
	second	0.318

4. Discussion

Overall, high performance was observed in the process mining stage. In the stage of classification, a high variance was seen between different decision points, both when

using different performance metrics and when using different configurations. We noticed a relatively low predictive performance at the initial decision point source0. An erroneous prediction at source0 sets an incorrect path for the next step in the traversal of the process model, resulting in lower Jaccard score.

Our methodology can be applied to multiple other domains, by creating a domain-specific temporal-abstraction knowledge base and then following the same steps in the process that we had described (Figure 1). Future research can include the use of additional temporal abstraction types and process mining algorithms to improve results.

5. Conclusions

In this study, we presented multiple contributions, including the representation of high-resolution longitudinal medical records in a process model, construction of the relevant datasets based on the process model, and per-decision-point classifier training using the datasets. We also demonstrated the ability to predict, to some extent, *all* of the events expected within a future time window, without being limited to one-step predictions.

An additional benefit of this work is the comparison of process mining to temporal data mining techniques that discover frequent temporal patterns, such as KarmaLego and similar approaches. KarmaLego highlights local behavior, and is more detailed and expressive, while process mining assesses large trajectories. Using both methods combined exploits both of their respective advantages, as demonstrated in this study.

Acknowledgements

This work was partially supported by ONR award No. N629091912124 and by the Israeli Medical Corps research program.

References

[1] Lee JM, Hauskrecht M. Recent Context-Aware LSTM for Clinical Event Time-Series Prediction. In: Riaño D, Wilk S, ten Teije A, editors. AIME [Internet]. Springer; 2019. p. 13–23. Available from: http://dblp.uni-trier.de/db/conf/aime/aime2019.html#LeeH19.

[2] Johnson A, Bulgarelli L, Pollard T, Horng S, Celi L A, Mark R. MIMIC-IV (version 2.2). PhysioNet. 2023, doi: https://doi.org/10.13026/6m.

[3] Weijters AJ, van Der Aalst WM, De Medeiros AA. Process mining with the heuristics miner-algorithm. Technische Universiteit Eindhoven, Tech. Rep. WP. 2006 Jan;166:1-34.

[4] Leemans SJJ, Fahland D, Aalst WMP van der. Discovering Block-Structured Process Models from Event Logs Containing Infrequent Behaviour. In: Lohmann N, Song M, Wohed P, editors. Business Process Management Workshops, International Workshop on Business Process Intelligence (BPI 2013). Springer-Verlag, Berlin; 2014. p. 66-78, doi: 10.1007/978-3-319-06257-0_6.

[5] Shahar Y. A framework for knowledge-based temporal abstraction. Artif intell. 1997 Feb;90(1-2):79-133, doi: 10.1016/S0004-3702(96)00025-2.

[6] Moskovitch R, Shahar Y. Fast time intervals mining using the transitivity of temporal relations. Knowl Inf Syst. 2015 Jan;42(1):21-48, doi: 10.1007/s10115-013-0707-x.

[7] Moskovitch R, Shahar Y. Classification of multivariate time series via temporal abstraction and time intervals mining. Knowl Inf Syst. 2015 Oct;45(1):35-74, doi: 10.1007/s10115-014-0784-5.

[8] Chen T, Guestrin C. XGBoost: A Scalable Tree Boosting System. In: Krishnapuram B, Shah M, Smola AJ, Aggarwal C, Shen D, Rastogi R, editors. KDD [Internet]. ACM; 2016. p. 785–94. Available from: http://dblp.uni-trier.de/db/conf/kdd/kdd2016.html#ChenG16.