


pMineR: An Innovative R Library for Performing Process Mining in Medicine

Roberto Gatta¹✉, Jacopo Lenkowicz¹, Mauro Vallati², Eric Rojas³,
Andrea Damiani¹, Lucia Sacchi⁴ , Berardino De Bari⁵, Arianna Dagliati⁶,
Carlos Fernandez-Llatas⁷, Matteo Montesi⁸, Antonio Marchetti⁸,
Maurizio Castellano⁹, and Vincenzo Valentini¹

¹ Università Cattolica del Sacro Cuore, Roma, Italy
`roberto.gatta.bs@gmail.com`

² University of Huddersfield, Huddersfield, UK

³ Universidad Pontificia Católica de Chile, Santiago, Chile

⁴ Università degli Studi di Pavia, Pavia, Italy

⁵ Centre Hospitalier Régional Universitaire de Besançon, Besançon, France

⁶ IRCCS ICS Maugeri Pavia, Pavia, Italy

⁷ ITACA-Universitat Politècnica de Valencia, Valencia, Spain

⁸ Data Warehouse, Policlinico Universitario A. Gemelli, Roma, Italy

⁹ Università degli Studi di Brescia, Brescia, Italy

Abstract. Process Mining is an emerging discipline investigating tasks related with the automated identification of process models, given real-world data (Process Discovery). The analysis of such models can provide useful insights to domain experts. In addition, models of processes can be used to test if a given process complies (Conformance Checking) with specifications. For these capabilities, Process Mining is gaining importance and attention in healthcare.

In this paper we introduce pMineR, an R library specifically designed for performing Process Mining in the medical domain, and supporting human experts by presenting processes in a human-readable way.

Keywords: Process mining · R · Decision support system

1 Introduction

Process Mining [1] is an emerging discipline aiming at providing methods to automatically identify, starting from real-world data or Event Logs, accurate models of processes. This is usually defined as Process Discovery.

Among the other domains, medicine is one of the most complex from the Process Mining perspective. This is due, for instance, to the high knowledge needed to deal with corresponding data, the need to involve experts –that usually are not strongly in favor of automatic techniques–, and the lack of formal methodologies. On this matter, the interested reader is referred to a recent review about the role of Process Mining in healthcare [5].

It should be noted that, at the state of the art, there are few tools that can perform Process Mining. Well-known examples include DISCO [2], PROM [6] and PALIA [2]. However, there is a lack of tools focusing on Process Mining in Medicine. Moreover, we observed that a common expectation of users from the medical domain, is to be able to work with one of the most diffused statistical software (like MATLAB, R, or SPSS), to exploit the previous knowledge of a well-known environment. R [4], in particular, is one of the most common software environment for data analysis: nevertheless, with the exception of *edeR* [3], which unfortunately does not support Process Discovery or Conformance Checking, there is no availability of libraries to cope with Process Mining.

In order to fill the aforementioned gap, in this work we propose *pMineR*,¹ an R library focused on dealing with Process Discovery and Conformance Checking in the medical domain. In such domain, it is of pivotal importance that physicians clearly understand every aspect of the elements in the generated models. In the light of that, *pMineR* exploits Markov Models, which are usually well-known and easy to understand for medical users.

While coping with Process Discovery, *pMineR* exploits some aspects taken from the Computer Interpretable Clinical-Guidelines field, in particular in terms of human-readability and representation of clinical guidelines. About this point big efforts have been made in proposing a new language, less powerful but easier to be handled from physicians to the extant ones.

2 The *pMineR* Library

The *pMineR* structure has been specifically designed for supporting: (i) the evolution and extensibility of the system, and (ii) to privilege improvements oriented to cope with real-world issues in medicine (it has to have a real-needs-driven improvement). The former element is reflected in a very modular architecture, and in the adoption of a strategy to govern architectural improvements by a set of development guidelines. The latter has been implemented by emphasizing a goal-driven development, by proposing scenarios of application, and by structuring the documentation in form of Case Reports.

The modular architecture exploits *collections*, specialized sets of classes built using common methods and exchanging data structures. This is done in order to maximize the re-usability of the code and the extensibility of *pMineR*.

The main *collections* of *pMineR* are described in the following.

- **Data Loader:** this *collection* includes classes handling the loading of event logs, and data pre-processing. It provides tools to translate and/or group event log terms by a given dictionary, to load arrays of *dataLoader* objects (used for storing structured information extracted from raw data) and manipulate them for creating different *views*.
- **LOG Inspection:** is a set of classes aiming to provide some descriptive statistics on event log data, such as events and processes, useful for a preliminary exploration of the data.

¹ <https://cran.r-project.org/web/packages/pMineR/index.html>.

- **Process Discovery:** classes in this *collection* implement one or more Process Discovery algorithms, given a *Data Loader* object. In the current version of pMineR there are two classes implementing, respectively, first and second order Markov Models-based algorithms. Classes in this package interact by a set of standard methods such as *load*, *train*, *play*, *replay*, in order to increase the possibility of interaction among objects of different classes.
- **Conformance Checking:** is a set of classes specialized in Conformance Checking. Even if also the Process Discovery classes have methods to check how a set of given processes can flow through the models, normally, the formalism for representing clinical guidelines have no algorithms to automatically generate an interpretable guideline starting from real world data: pMineR implements a class able to work with an internal formalism for representing Workflow-like diagrams: such formalism is called Pseudo-Workflow (PWF) and was designed with the contribution of our physicians in order to be human readable and to be able to represent a set of needed guidelines.
- **General Purposes:** this *collection* include classes addressing a wide range of common issues that are faced during programming, such as exception handling, and strings manipulation.

2.1 Process Discovery

The current version of pMineR implements two algorithms referring to the first (FOMM) and second (SOMM) order Markov Models. Both of them provide the minimum set of common methods and some different model-specific methods.

Some of the methods provided by classes of the Process Mining package are the common expected methods of PM algorithms, such as: *load*, *train*, *play*, and *replay*, which role is well described in the Manifesto [1]. Some more advanced methods include *compare*, which can be used to compare two models given an embedded default metric or a user-defined one (the simplest default metric for FOMM model is the normalized sum of absolute delta between two FOMMs). pMineR can compute the differences between generated models, and output them under the form of diagrams, that can then be analyzed by human experts.

Figure 1 (colored) shows the difference among two FOMM models, built using the *compare* method.

2.2 Conformance Checking

These classes are specifically designed to support Conformance Checking, by proposing schema and diagrams close to the language adopted by physician. At the moment, pMineR implements an engine which is able to parse guidelines written in the previously introduced PWF language. PWF is based on three main constructs: events, statuses, and triggers.

Given an event log, the engine reads the list of events and, for each event, it tests if one or more triggers can be fired. A trigger is an item composed by two main sections: condition and effects. The condition part can check elements of the just read event log (*\$ev.NOW\$*) or other statuses of the patient (e.g.,

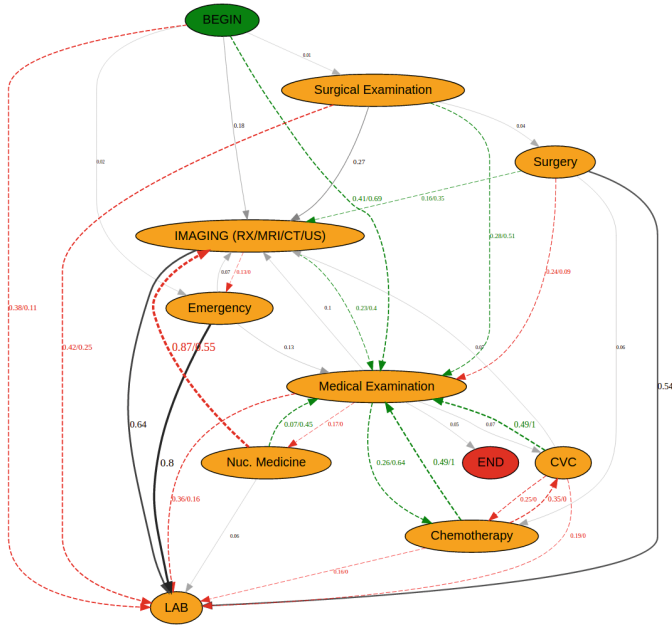


Fig. 1. The graph shows a FOMM related to a set of patient treated for lung cancer, compared with a set of patient treated for breast cancer: in such graph the arcs in black are the arch with a similar probability (according to a threshold). Red (green) arcs indicate that the transition probability for the lung model is lower (higher) than the one for breast. (Color figure online)

currently active statuses $\$st.ACTIVES\$$). If the condition applies, the effects listed in the subsequent section are executed. In the following example, a trigger for representing the end of a treatment is specified, using the PWF language. If the current status of the treatment is *in progress*, and a *dismissal Report* event is read, the status of the patient has to be updated, according to the list of **set** and **unset** items. Using this approach, statuses are automatically updated while events are processed, sequentially, from the first to the last.

Figure 2 provides an example of the computation of a PWF for a dummy set of event log (on the left) and details about a specific patient (on the right). On the left the workflow is graphed starting from the given XML used for defining triggers (squared boxes) and statuses (rounds). On the top right an original event log, which is an input of the computation. On the bottom right, the result of the computation for the same event log, plotted under the form of the “activation time” of the different statuses.

In addition, PWF also provides a set of items for representing time and duration. It can also support different kind of *actions* on status, and allows to specify a header, on the guidelines, to indicate references, authors, validity, and information about the expected use.

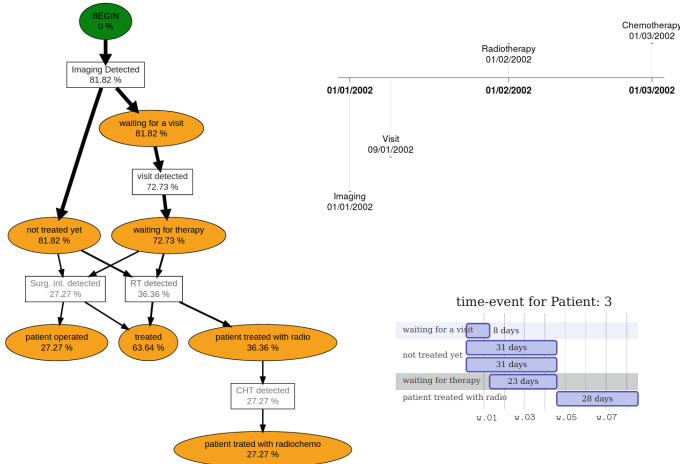


Fig. 2. Example output provided by pMineR after the computation of a PWF.

3 Conclusion

While several tools have been developed for dealing with the general Process Mining problem, only a few are able to deal with some aspects of its application to the medicine domain. Moreover, there is a lack of libraries for Process Mining in R, one of the most exploited software environment for data analysis. For filling these gaps, in this paper we presented pMineR, an innovative R library addressing all the tasks of Process Mining, with a specific focus on the medical domain. It should be noted that, while developing pMineR, we took into account insights provided by the state of the art of Computer Interpretable clinical Guideline (CIG), in order to provide easy to understand and interpret output. The system is already freely available, and can be exploited for efficiently and effectively performing Process Mining on sets of heterogeneous real world data.

References

1. Aalst, W., et al.: Process mining manifesto. In: Daniel, F., Barkaoui, K., Dustdar, S. (eds.) BPM 2011. LNBP, vol. 99, pp. 169–194. Springer, Heidelberg (2012). doi:10.1007/978-3-642-28108-2_19
2. Günther, C.W., Rozinat, A.: Disco: discover your processes. BPM (Demos) **940**, 40–44 (2012)
3. Janssenswillen, G.: edeaR. cran.r-project.org/web/packages/edear
4. R Core Team: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing (2016). R-project.org
5. Rojas, E., Munoz-Gama, J., Sepúlveda, M., Capurro, D.: Process mining in health-care: a literature review. J. Biomed. Inform. **61**, 224–236 (2016)
6. Dongen, B.F., Medeiros, A.K.A., Verbeek, H.M.W., Weijters, A.J.M.M., Aalst, W.M.P.: The ProM framework: a new era in process mining tool support. In: Ciardo, G., Darondeau, P. (eds.) ICATPN 2005. LNCS, vol. 3536, pp. 444–454. Springer, Heidelberg (2005). doi:10.1007/11494744_25