From low-level-event-logs to high-level-business-process-model-activities: An advanced framework based on machine learning and flexible BPMN model translation

From Low-Level-Event-Logs to High-Level-Business-Process-Model-Activities: An Advanced Framework based on Machine Learning and Flexible BPMN Model Translation

Alfredo Cuzzocrea[1], Ernesto Damiani[2], Hamda Al-Ali[3], Rabeb Mizouni[4], Ghalia Tello[5] and Edoardo Fadda[6]

[1]*iDEA Lab, University of Calabria, Rende, Italy*
[2]*Khalifa University & EBTIC, Abu Dhabi, UAE*
[3]*Khalifa University, Abu Dhabi, UAE*
[4]*Khalifa University, Abu Dhabi, UAE*
[5]*Khalifa University, Abu Dhabi, UAE*
[6]*Politecnico di Torino & ISIRES, Torino, Italy*

Abstract

Process mining is an emerging discipline that aims to analyze business processes using event data logged by IT systems. In process mining, the focus is on how to effectively and efficiently predict the next process/trace to be activated among all the possible processes/traces that are available in the process schema (usually modeled as a graph). Most of the existing process mining techniques assume that there is a one-to-one mapping between process model activities and the events that are recorded during process execution. However, event logs and process model activities are at different level of granularity. In this paper, we present a machine learning-based approach to map low-level event logs to high-level activities. With this work, we can bridge the abstraction levels when the high-level labels of the low-level events are not available. The proposed approach consists of two main phases: automatic labeling and machine learning-based classification. In automatic labeling a modified k- prototypes clustering approach has been used in order to obtain the labeled examples. Then, in the second phase, we trained different ML classifiers using the obtained labeled examples. Since, in real-life applications and systems, business processes are expressed according to the Business Process Model and Notation (BPMN) format, we improve our proposed framework by means of an innovative, flexible BPMN model translation methodology that acts at the first phase.

Keywords

Business process management, Business process mining, BPMN model translation

1. Introduction

With the advent of Information technology (IT), companies and organizations adopt IT services to model and execute their business processes. A shift from data orientation to process orienta-

SEBD 2021: The 29th Italian Symposium on Advanced Database Systems, September 5-9, 2021, Pizzo Calabro (VV), Italy
alfredo.cuzzocrea@unical.it (A. Cuzzocrea); ernesto.damiani@kustar.ac.ae (E. Damiani); 100035242@kustar.ac.ae (H. Al-Ali); rabeb.mizouni@kustar.ac.ae (R. Mizouni); ghalia.tello@kustar.ac.ae (G. Tello); edoardo.fadda@polito.it (E. Fadda)

tion has been witnessed by the last decades [1]. A business process intends to break up a certain job into sequence of activities, where different individuals can be responsible for different activities. Managing these business processes in an accurate, efficient, and well-organized way has a vast influence on the organization output and so its success. Accordingly, Business Process Management (BPM) became a vital discipline to organizations in order to model, discover, analyze and improve their business processes. Process mining is an emerging research area in the field of process management, which focuses on extracting process related information using event data logged by the IT systems. Process mining techniques can automatically discover process models, check the conformance of process execution to model specification, and enhance existing process models [2, 3].

Recently, an interesting synergy between business processes and emerging *big data management and processing* (e.g., [4, 5, 6, 7, 8, 9, 10, 11]) has been highlighted by several studies, among which noticeable ones are: [12, 13, 14, 15]. Looking at the actual literature, there are several initiatives about on how to shape the process mining framework, particularly for what regards the mapping between event log database and the sovereign process model. First, we focus on one-to-one mapping approaches. Most of these approaches assume that there is a one-to-one mapping between events in the log and process model activities. One-to-one mapping can be achieved by simple string substitution of the event names with activities names. However, event logs and process model activities are at different level of granularity [16]. Specifically, events in the log are finer grained than activities in process models. Therefore, an efficient algorithm for mapping low-level events to high-level activities is required in order to enable process mining techniques such as conformance checking and model enhancement. Moreover, this mapping is important for process discovery algorithms in order to discover more representative and interpretable process models. Without this mapping, the discovered models might be too complicated with too specific and non-meaningful activity names. On the other hand, many-to-many mapping approaches try to recover the activity life-cycle from a set of correlated event in the trace of the log. This model better capture real-life systems. Our proposed framework, indeed, fits well right in the direction of hybrid approaches, where the goal is, as mentioned, bridging the gap between activity logs and business process management.

In this paper, we design and implement an artificial intelligence model that is able to learn the mapping between low-level event logs and process model activities. This model comprises two main phases. The first phase is an automatic labeling approach. The goal of this phase is to automatically assign sequences of low-level events with high-level target labels. This phase is crucial since the labels for log traces are usually unavailable, and manual labeling might be infeasible, time consuming or too expensive. In the second phase, a supervised Machine Learning (ML) classifier is used to learn the mapping between low-level event logs and model activities. The labeled examples generated by the first phase are used to train the supervised ML classifier of the second phase. Since, in real-life applications and systems, business processes are expressed according to the Business Process Model and Notation (BPMN) format, we improve our proposed framework by means of an innovative, flexible BPMN model translation methodology that acts at the first phase. In particular, the need to supporting BPMN translation to BP rules is necessary as we need to "reduce" the complex BPMN models to lower-level rules to be included in the machine learning phase directly. With this work, we can automatically and accurately bridge the abstraction levels when target labels are not available. Moreover, most of the existing

abstraction approaches aim to support model discovery techniques in order to discover more interpretable process models. However, we aim to enable conformance checking techniques by getting a high accuracy of mapping event logs to existing activities in the process model.

Under a broader umbrella, our framework is focused to support conformance checking of business processes. Indeed, the final goal of our framework is to support the mapping between low-level event logs and high-level business processes model activities, thus implementing the conformance checking between the two components of any arbitrary business process system. It should be noted that this requirement is extremely important in a wide range of actual application scenarios, ranging from traditional banking/assurance systems to emerging industry 4.0 settings. Conformance checking falls under the umbrella of *anomaly detection* because it provides information about mismatches (often called *violations*) between logs and process models and helps process owners to understand the causes. Business rules define constrains or guidelines that apply to an organization. Organizations use business rules to enforce policy, comply with legal obligations, communicate between various parties and perform process analysis.

In particular, we deal with process-specific rules, i.e. rules that constrain the behavior of a business process in order to achieve a specific goal. These rules can be hidden in source code, inside use cases or in workflow descriptions [17]. In this work, we use a simple language to extract logic constraints directly from BPMN models and then translate them into business rules. Using the language, any user can extract the rules easily since they are simple and tool-independent. We build a prototype to extract the logic rules automatically using the XML schema of the BPMN model. Our approach builds on previous work proposed by [18]. In their paper, the authors introduced a mechanism to translate BPMN models to Business Process Execution Language (BPEL), our technique, instead, focuses on the translation of BPMN model to business rules[1].

2. Overview on the Proposed Framework

The proposed framework is summarized in Fig. 1. The overall approach can be subdivided into two phases. The first phase is an automatic labeling task. The high-level target labels should be available in order to learn the mapping between low-level events and high- level activities. Manual labeling is time consuming or even infeasible [19]. Therefore, the automatic labeling module is used to create labeled examples. These labeled examples are refined by taking only a subset of examples that contribute to the goodness of the labeling task. With the aim of improving the effectiveness and the quality of the proposed framework, since, in real-life applications and systems, business processes are usually expressed in terms of BPMN models, we include an innovative, flexible BPMN model translation methodology that allows us to translate such models in suitable BP rules. Indeed, as highlighted in Section 1, the need to supporting BPMN translation to BP rules is necessary as we need to "reduce" the complex BPMN models to lower-level rules to be included in the machine learning phase directly. In the second phase, the labeled examples are used to train a machine learning classifier. The classifier will learn the mapping between low-level sequences and high-level activities. After learning the

[1]Some definitions used in our work are taken verbatim from their paper.

mapping, the classifier will be able to map a new set of unlabeled low-level sequences of events to the corresponding high-level activities. Hence, a high-level activity log can be created.

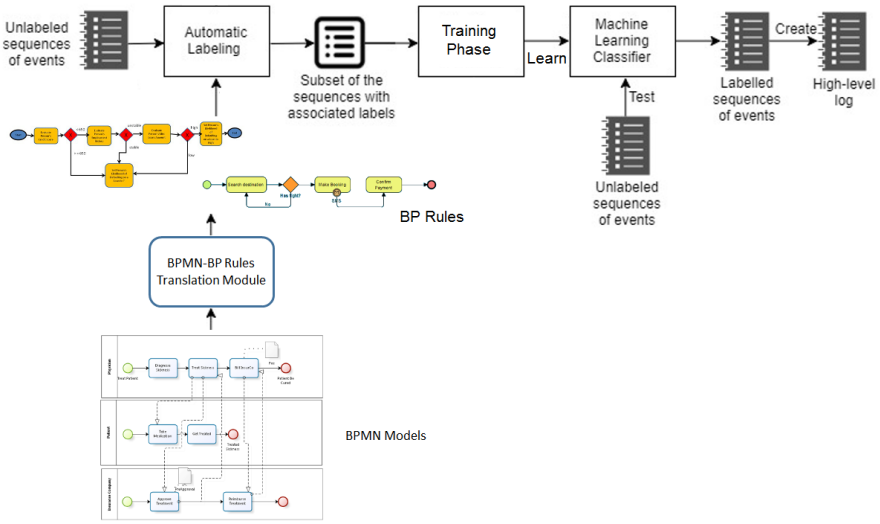

Figure 1: Overview of the proposed framework

As shown in Fig. 1, the proposed framework is composed by the following entities and (software) modules:

- *Unlabeled sequences of events* – these are low-level events from which the framework finally extracts the high-level business process activities;
- *BPMN Models* – these are the BPMN models of the reference BPMN; they are exploited by our translation module in order to improve the overall effectiveness and the quality of the proposed framework;
- *BPMN-BP Rules Translation Module* – it is the module for supporting the translation from BPMN models to BP rules, in a flexible manner;
- *Automatic Labeling* – it is the module that is in charge of supporting the creation of labeled examples to be provided as input to the supervised machine learning classification phase;
- *Subset of the sequences with associated labels* – these are the sequences extracted from the automatic labeling phase, which embed associated labels;
- *Unlabeled sequences of events* – similarly to the main input, these are low-level events that are processed by the machine learning classification phase;
- *Machine Learning Classifier* – it is the main component where the machine learning classification phase is performed in order to finally extract the high-level business process activities;
- *Labeled sequences of events* – these are the output of the machine learning classification phase; (classification) labels are used to finally build the desired high-level business process activities;

- *High-level logs* – these are the final output of the proposed framework: high-level business process activities constructed from the low-level events directly.

3. Translating BPMN to Business Process Rules

In the BPMN-BP rules translation phase, we introduce a simple, human-readable rule language based on a fragment of First-Order Logic (FOL) and show how compliance rules can be generated directly from BPMN models. We focus on control flow aspects of BPMN models by (1) transforming the model to obtain a uniform representation of task activation (2) dividing the model into sets of components and (3) using our proposed language to generate compliance rules for each component. We show that these rules can be used in the analysis of the business process execution log using British Telecom's Aperture business process analysis tool.

We start by extracting FOL constraints directly from BPMN models. Our constraints will be translated later into business rules. We rely on an initial graph transformation to achieve an implicit uniform task activation semantics; then, we apply the basic definitions given in [18] with some minor variations [2].

3.1. Graph Transformation

We start by translating the BPMN model into a fully synchronous workflow. In BPMN, activity are by default performed synchronously in relation to the invoking process flow, i.e. the process waits for an activity to complete before the process can proceed. However, BPMN syntax allows specifying asynchronous activity execution, e.g. requiring an external event to take place for enabling the execution of an activity. Using asynchronous events (rather than the completion of a previous activity) to enable execution of activities provides a general way to express different enabling semantics. The gist of our transformation is to avoid this complexity by treating synchronization events as special case of ordinary activities, and always use activity enabling by-compilation (of previous activity). In other words, before any analysis, all intermediate events in a BPMN model are transformed to special tasks with double borders to distinguish them. While such transformation may decrease the expressive power of the language, it has the advantage of decreasing the complexity of the model. For the exclusive gateway (XOR), we exclude the default statement, which leads to nothing. The WHILE component will be replaced with REAPEAT to avoid null activity. Summarizing, we perform the following transformations of the BPMN model:

1. ***step 1.*** *Conversion of events into activities*;
2. ***step 2.*** *Elimination of DEFAULT in XOR component*;
3. ***step 3.*** *Substitution of WHILE with REPEAT component.*

As shown in Fig. 2, the intermediate event e_1 is transformed to a task t_2. The DEFAULT sequence flow in the switch component is removed. At the end, the WHILE component is substituted with REPEAT component.

[2]The full definitions can be found in [18]

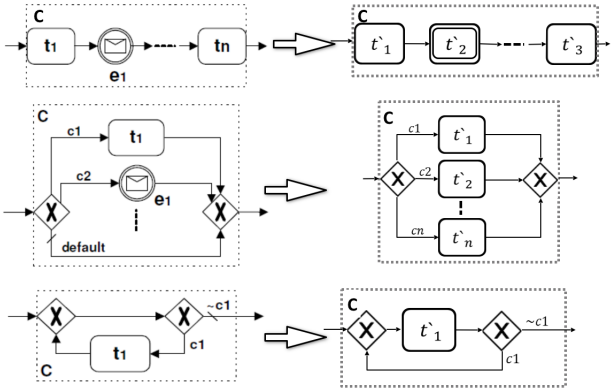

Figure 2: Intermediate event, elimination of DEFAULT and substitution of WHILE transformation

3.2. Business Process Diagram (BPD)

Business processes are expressed graphically using BPMN elements in a BPD. The model is composed of a set of different tasks, events and gateways referred as objects. A task is a single activity in the model while events can represent the start, intermediate, end, and termination of the process (graph transformation will exclude intermediate events), While the gateway represents parallel and XOR forks and joins. Fig. 3 shows the graphical representation of some BPMN elements in a core BPD which is composed of set of objects that can be partitioned into disjoint sets of tasks $\mathcal{T}$, events $\mathcal{E}$ and gateways $\mathcal{G}$ [18].

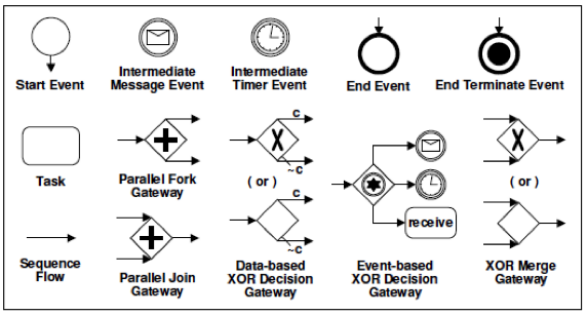

Figure 3: A core subset of BPMN elements [18]

In the remainder of the paper, we only consider well-formed core $\mathcal{BPD}s$ as defined in [18]. Moreover, without losing generality we assume that both $\mathcal{E}^{\mathcal{S}}$ *and* $\mathcal{E}^{\mathcal{E}}$ *are singletons, i.e.* $\mathcal{E}^{\mathcal{S}} = \{s\}$ *and* $\mathcal{E}^{\mathcal{E}} = \{e\}$.

3.3. Decomposing a BPD into Components

The notion of component is used to transform a graph structure into set of business rules. To facilitate this transformation, the BPD is decomposed into different components. Again

according to [18] "A component is a subset of the BPD that has one entry and one exit point". Each component will be mapped into a single logic rule. Each component should include a minimum number of two different objects (source and sink). A BPD with no component which only contain a single task between the start and end events is called a *trivial* BPD. Whenever we reach a *trivial* BPD, no rule can be extracted and therefore we stop the translation. Breaking down the BPD into set of components helps to define an iterative method to transform BPD into rules. A function **Fold** is defined in [18] which substitutes a component with single task. **Fold** function can be utilized to reduce the BPD iteratively until we reach a *trivial* BPD.

3.4. Structured Activity-Based Translation

In our approach, different components are mapped into a subset of FOL rules including AND, XOR and sequence operations. Paper [18] defines seven forms of well-structured components. Fig. 4 represents the mapping of each component into the corresponding FOL rules [18].

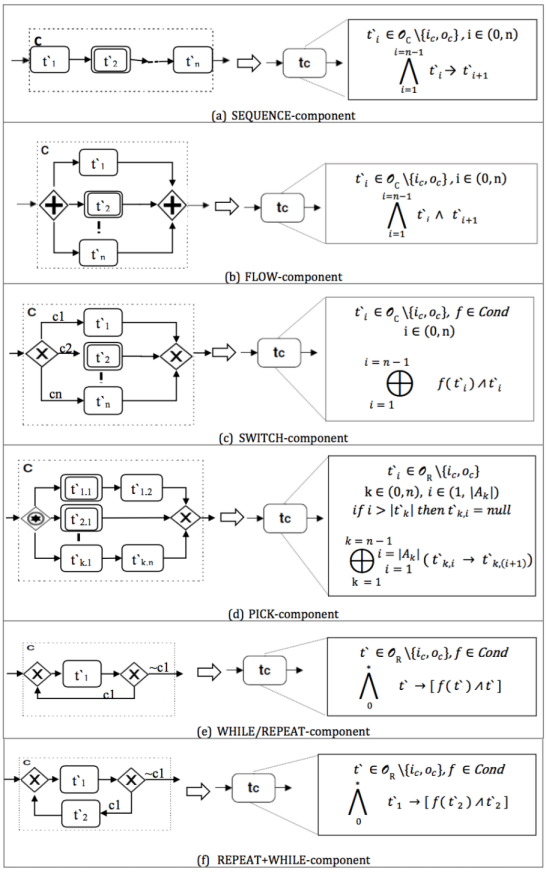

Figure 4: Folding a well-structured component $\mathscr{C}$ into single task object t_c attached with the corresponding FOL rule translation of $\mathscr{C}$ [18]

Each rule corresponds to a specific position in the BPD. The position information can be utilized in different ways in the conformance checking process and introduces two different

types of dependencies: sequential and hierarchical dependencies. Sequential order means rules extracted from earlier components should be checked before rules from later components. It should be noted that sequential order is the most "natural" way to derive the corresponding BP rules, as it completely adheres to the the FOL that is at the basis of the entire process. Indeed, at a practical level, this is also more convergent to the "intuitive" folding of structural components (i.e., components that are composed by a collection of basic components), which is the most convenient approach for real-life complex business process management systems.

On the other hand, this technique presents the notion of hierarchy of constraints, which to the best of our knowledge, is not well found in the literature. One or more rules can depend on another rule and therefore executing high-level constraints plays critical role in the execution of other low-level constraints. It should be noted that the hierarchical order is instead prone to capture even more complex real life settings where BPMN schema model emerging applications like e-government or e-procurement procedures (e.g., [20, 21, 22]). In fact, in such settings, it is easy to recognize a hierarchical structure of components into sub-components, which demand for different folding strategies.

3.5. Translation Algorithm

After mapping each component to the corresponding rule, we introduce the algorithm used to translation a well-formed core BPD into FOL rule which is similar to the algorithm introduced in [18] with some modifications. The algorithm includes three different steps, selecting a well-structured component then providing its FOL rule and finally fold the component. This is done repeatedly until we reach a *trivial* BPD.

[**Algorithm 1**[18]]

Let $\mathscr{BPD} = (\mathscr{O}, \mathscr{F}, Cond)$ be a well-formed core $\mathscr{BPD}$ with one start event and one end event. $[X]_c$ is the set of components of BPD[X].

1. $X := \mathscr{BPD}$
2. if $[X]_c = \emptyset$ (i.e., X is initially a *trivial*$\mathscr{BPD}$), stop.
3. while $[X]_c \neq \emptyset$ (i.e., X is a non-trivial $\mathscr{BPD}$)
   a) if there is a maximal SEQUENCE component $\mathscr{C} \in [X]_c$, select it and goto (3-c).
   b) if there is a well-structured (non-sequence) component $\mathscr{C} \in [X]_c$, select and goto (3-c).
   c) Attach logic rule translation of $\mathscr{C}$ to task object t_c.
   d) $X :=$ **Fold**$(X, \mathscr{C}, t_c)$ and return to (3).
4. Output the logic rule attached to the task object t_c.

4. Automatic Labeling

The goal of this phase is to create labeled examples to feed the supervised machine learning classifier of the next phase. In the process mining field, the execution of processes is reflected by event logs. The eXtensible Event Stream (XES) defines a standard for recording information system's events. Typically, an event in the log is defined as $e = (c, n, r, t)$, representing the occurrence of an event n, in a case c, using the resource r , at time-stamp t [23]. Additional

attributes for the events can be included such as price, originator, location, etc. As highlighted in Section 2, the automatic labeling phase takes advantages from the BPMN-BP Rules Translation Module, in order to discover useful business rules hidden in BPMN models.

In the automatic labeling module, we followed a clustering approach to cluster the multivariate time series log sequences with numerical and categorical attributes. Different clustering algorithms can be followed to achieve this task. In the following, we specifically focus on such clustering tools.

When implementing automatic labeling based on clustering approaches, the general architecture of the proposed framework depicted in Fig. 1, such architecture specifies as reported in Fig. 5. The overall approach can be subdivided into two phases. The first phase is a clustering-based labeling approach. Our framework is not strictly constrained to a particular clustering algorithm. Indeed, in this phase different clustering algorithms can be used, depending on the particular application setting considered (in fact, we later discuss the specific clustering algorithm selected in our implementation). The "orthogonality" of the clustering algorithm should be intended as another amenity of our proposed framework. The high-level target labels should be available in order to learn the mapping between low-level events and high- level activities. Manual labeling is time consuming or even infeasible [19]. Therefore, the labeling module is used to create labeled examples. For every activity, a cluster of sequences of events should be formed. And then, all samples will be labeled with same labels as their corresponding clusters. After that, these labeled examples are refined by taking only a subset of examples that contribute to the goodness of the labeling task. In the second phase, the labeled examples are used to train a machine learning classifier. The classifier will learn the mapping between low-level event sequences and high-level activities. After learning the mapping, the classifier will be able to map a new set of unlabeled low-level sequences of events to the corresponding high-level activities. Hence, a high-level activity log can be created.

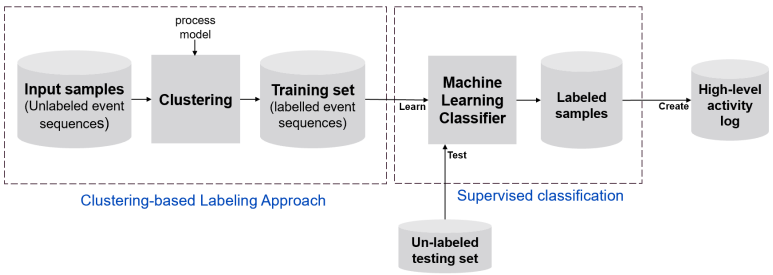

Figure 5: Overview of the proposed framework when implementing automatic labeling based on clustering approaches

4.1. Clustering-Based Labeling Approach

The first phase is the clustering-based labeling approach, which is illustrated in Fig. 6. The goal of this phase is to create labeled examples to feed the supervised machine learning classifier of the next phase.

The input samples of this phase are unlabeled event sequences. Each sample is a set of

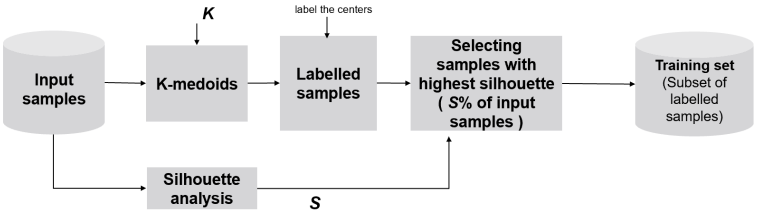

Figure 6: Clustering-based Labeling approach being K the number of high-level activities from the high-level process model, and S the percentage of samples with highest silhouette

low-level events. In order to provide labels for these samples, we will cluster them into different clusters, where each cluster represents a high-level activity. Different clustering algorithms can be followed to achieve this task. As discussed in Section 4, the specific clustering algorithm is orthogonal to the proposed framework. In our implementation, we selected k-medoids (e.g., [24]) as main clustering algorithm.

In particular, we use two clustering algorithms based on the attribute types of the input samples. Specifically, k-prototypes is utilized to cluster multivariate time series log sequences with numerical and categorical attributes, while k-medoids is applied to cluster samples with categorical attributes. Furthermore, clustering-based approach can know from the high-level process model the number of high-level activities in the process, which accordingly represents the number of clusters that should be formed. Each sample within one cluster will be labeled with its cluster label. In fact, in order to label the clusters, the domain expert should label only the cluster centers, and then each cluster label will be the same as the label of it's center.

In the process mining field, the execution of processes is reflected by event logs. The eXtensible Event Stream (XES) defines a standard for recording information system's events. Typically, an event in the log is defined as $e = (n, c, r, s, t)$, representing the occurrence of an event n, in a case c, using the resource r, having a status $s \in \{\text{start, complete}\}$, at time-stamp t [23]. Additional attributes for the events can be included such as price, originator, location, etc.

5. Conclusions and Future Work

In this paper, we proposed a ML-based approach to bridge the gap between low-level event logs and high-level activities when high-level labels of the low-level events are not available. Our proposed method consists of two main phases: clustering-based labeling approach and supervised ML-based classification. Since, in real-life applications and systems, business processes are expressed according to the BPMN format, we improved our proposed framework by means of an innovative, flexible BPMN model translation methodology that acts at the first phase. Future work is oriented towards embedding adaptive metaphors to our proposed framework, even inherited by different-but-related contexts (e.g., [25]).

References

[1] M. Dumas, W. M. Van der Aalst, A. H. Ter Hofstede, Process-aware information systems: bridging people and software through process technology, John Wiley & Sons, 2005.

[2] W. M. P. van der Aalst, Process Mining - Discovery, Conformance and Enhancement of Business Processes, Springer, 2011.

[3] A. Augusto, R. Conforti, M. Dumas, M. L. Rosa, F. M. Maggi, A. Marrella, M. Mecella, A. Soo, Automated discovery of process models from event logs: Review and benchmark, IEEE Trans. Knowl. Data Eng. 31 (2019) 686–705.

[4] A. Cuzzocrea, Improving range-sum query evaluation on data cubes via polynomial approximation, Data Knowl. Eng. 56 (2006) 85–121.

[5] P. Braun, J. J. Cameron, A. Cuzzocrea, F. Jiang, C. K. Leung, Effectively and efficiently mining frequent patterns from dense graph streams on disk, in: KES, volume 35 of *Procedia Computer Science*, Elsevier, 2014, pp. 338–347.

[6] C. Yang, J. Liu, C. Hsu, W. Chou, On improvement of cloud virtual machine availability with virtualization fault tolerance mechanism, The Journal of Supercomputing 69 (2014) 1103–1122.

[7] K. Li, H. Jiang, L. T. Yang, A. Cuzzocrea (Eds.), Big Data - Algorithms, Analytics, and Applications, Chapman and Hall/CRC, 2015.

[8] A. Cuzzocrea, E. Bertino, Privacy preserving OLAP over distributed XML data: A theoretically-sound secure-multiparty-computation approach, J. Comput. Syst. Sci. 77 (2011) 965–987.

[9] A. Cuzzocrea, V. Russo, Privacy preserving OLAP and OLAP security, in: Encyclopedia of Data Warehousing and Mining, IGI Global, 2009, pp. 1575–1581.

[10] A. Cuzzocrea, R. Moussa, G. Xu, Olap*: Effectively and efficiently supporting parallel OLAP over big data, in: MEDI, volume 8216 of *Lecture Notes in Computer Science*, Springer, 2013, pp. 38–49.

[11] A. Campan, A. Cuzzocrea, T. M. Truta, Fighting fake news spread in online social networks: Actual trends and future research directions, in: 2017 IEEE International Conference on Big Data, BigData 2017, Boston, MA, USA, December 11-14, 2017, IEEE Computer Society, 2017, pp. 4453–4457.

[12] L. Dezi, G. Santoro, H. Gabteni, A. C. Pellicelli, The role of big data in shaping ambidextrous business process management: Case studies from the service industry, Business Proc. Manag. Journal 24 (2018) 1163–1175.

[13] G. Festa, I. Safraou, M. T. Cuomo, L. Solima, Big data for big pharma: Harmonizing business process management to enhance ambidexterity, Business Proc. Manag. Journal 24 (2018) 1110–1123.

[14] R. Rialti, G. Marzi, M. Silic, C. Ciappei, Ambidextrous organization and agility in big data era: The role of business process management systems, Business Proc. Manag. Journal 24 (2018) 1091–1109.

[15] S. F. Wamba, Big data analytics and business process innovation, Business Proc. Manag. Journal 23 (2017) 470–476.

[16] T. Baier, J. Mendling, Bridging abstraction layers in process mining: Event to activity mapping, in: Enterprise, Business-Process and Information Systems Modeling, Springer, 2013, pp. 109–123.

[17] B. von Halie, L. Goldberg (Eds.), The Business Rule Revolution, Happy About, 2006.

[18] C. Ouyang, W. M. van der Aalst, M. Dumas, A. H. ter Hofstede, Translating bpmn to bpel, 2006.

[19] N. Tax, N. Sidorova, R. Haakma, W. M. van der Aalst, Event abstraction for process mining using supervised learning techniques, in: Proceedings of SAI Intelligent Systems Conference, Springer, 2016, pp. 251–269.

[20] A. A. Kalenkova, A. A. Ageev, I. A. Lomazova, W. M. P. van der Aalst, E-government services: Comparing real and expected user behavior, in: Business Process Management Workshops - BPM 2017 International Workshops, Barcelona, Spain, September 10-11, 2017, Revised Papers, 2017, pp. 484–496.

[21] S. Alfadhel, S. Liu, F. O. Oderanti, Business process modelling and visualisation to support e-government decision making: Business/is alignment, in: Third International Conference, ICDSST 2017, Namur, Belgium, May 29-31, 2017, Proceedings, 2017, pp. 45–57.

[22] R. Costa, O. Garcia, M. J. Nuñez, P. M. N. Maló, R. Jardim-Gonçalves, Integrated solution to support enterprise interoperability at the business process level on e-procurement, in: Proceedings of the 3th International Conference on Interoperability for Enterprise Software and Applications, IESA 2007, March 27-30, 2007, Funchal, Madeira Island, Portugal, 2007, pp. 89–100.

[23] P. Ceravolo, E. Damiani, M. Torabi, S. Barbon, Toward a new generation of log pre-processing methods for process mining, in: International Conference on Business Process Management, Springer, 2017, pp. 55–70.

[24] H. Park, C. Jun, A simple and fast algorithm for k-medoids clustering, Expert Syst. Appl. 36 (2009) 3336–3341.

[25] M. Cannataro, A. Cuzzocrea, C. Mastroianni, R. Ortale, A. Pugliese, Modeling adaptive hypermedia with an object-oriented approach and XML, in: Proceedings of the Second International Workshop on Web Dynamics, WebDyn@WWW 2002, Honululu, HW, USA, May 7, 2002, volume 702 of *CEUR Workshop Proceedings*, CEUR-WS.org, 2002, pp. 35–44.