Explainable Predictive Process Monitoring

Riccardo Galanti[*‡], Bernat Coma-Puig[†], Massimiliano de Leoni[‡], Josep Carmona[†], and Nicolò Navarin[‡]
[*]myInvenio, Reggio Emilia, Italy, [‡]University of Padua, Padua, Italy, [†]Universitat Politècnica de Catalunya, Barcelona, Spain
Email: riccardo.galanti@my-invenio.com, {deleoni,nnavarin}@math.unipd.it
{bcoma,jcarmona}@cs.upc.edu

Abstract—**Predictive Business Process Monitoring is becoming an essential aid for organizations, providing online operational support of their processes. This paper tackles the fundamental problem of equipping predictive business process monitoring with explanation capabilities, so that not only the *what* but also the *why* is reported when predicting generic KPIs like remaining time, or activity execution. We use the game theory of Shapley Values to obtain robust explanations of the predictions. The approach has been implemented and tested on real-life benchmarks, showing for the first time how explanations can be given in the field of predictive business process monitoring.**

I. Introduction

Within the field of Process Mining, predictive monitoring aims to forecast the running process instances with the purpose of timely signalling those that require special attention (those that may take too long, cost too much, not be satisfactory, etc.). Several approaches have been proposed in literature to deal with predictive monitoring (cf. Section III-A and the survey by Márquez et al. [9]), which has received significant attention in the last years. However, the majority of these approaches rely on black-box models (e.g. based on LSTM, i.e. Long Short-Term Memory neural models), which are proven to be more accurate, at the cost of being unable to provide a feedback to the user. On the other hand, approaches based on explicit rules (e.g. based on classification/regression trees) tend to be significantly less accurate. While the priority remains on giving accurate predictions, users need to be provided with an explanation of the reason why a given process execution is predicted to behave in a certain way. Otherwise, users would not trust the model, and hence they would not adopt the predictive-monitoring technology [13], [4].

This paper tackles the problem of equipping process monitoring with explanations of the predictions. It leverages on current state of the art of Explainable AI (cf. Section III-B), defining a framework for explainable process monitoring of generic KPIs. The proposed framework is independent of the machine- or deep-learning technique that is employed to make the predictions. However, we aim to instantiate the framework to prove its effectiveness. With this aim in mind, we built a process-monitoring framework, based on LSTM models, that is also able to explain any generic KPI, numerical or nominal. In a nutshell, given a running case, our framework estimates the future KPI value and returns the set of attributes that influence its prediction the most.

Experiments were conducted on different benchmarks, including the real-life process of an Italian financial institute, with the aim of predicting different KPIs, namely remaining time, costs, and the eventual occurrence of certain undesired activities. Explanations can be generated at LSTM-model level, to be provided to process stakeholders to understand the general trend of the model, but also at run-time, to explain the predictions of each single running case. The explanations obtained for the aforementioned financial institute are in line with those of the analysts of the process. The remarkable difference is that our results were obtained within a few days of automatic computations, instead of long analyses.

The rest of the paper is organized as follows. Section II states the problem addressed in this paper. Section III summarizes the most relevant work related to process predictive monitoring and Explainable AI. Section IV sketches the state of the art on using LSTM models for predictive monitoring, on which we build to provide explanations. Section V reports on our framework for explainable predictive process monitoring. Section VI reports on our framework's operationalization, and on the case studies conducted with an Italian financial institute, whereas Section VII concludes the paper.

II. Problem Statement

The starting point for a prediction system is an *event log*. An event log is a multiset of *traces*. Each trace describes the life-cycle of a particular *process instance* (i.e., a *case*) in terms of the *activities* executed and the process *attributes* that are manipulated.

Definition 2.1 (Events): Let $\mathcal{A}$ be the set of process attributes. Let $\mathcal{W}_{\mathcal{A}}$ be a function that assigns a domain $\mathcal{W}_{\mathcal{A}}(a)$ to each process attribute $a \in \mathcal{A}$. Let be $\overline{\mathcal{W}} = \cup_{a \in \mathcal{A}} \mathcal{W}_{\mathcal{A}}(a)$. An *event* $e \in \mathcal{E}$ is a partial function $\mathcal{A} \not\rightarrow \overline{\mathcal{W}}$ assigning values to process attributes, with $e(a) \in \mathcal{W}_{\mathcal{A}}(a)$.
Note that the same event can potentially occur in different traces, namely attributes are given the same assignment in different traces. This means that potentially the entire same trace can appear multiple times. This motivates why an event log is to be defined as a multiset of traces.[1]

Definition 2.2 (Traces & Event Logs): Let $\mathcal{E}$ be the universe of events. A trace σ is a sequence of events, i.e. $\sigma \in \mathcal{E}^*$. An event-log L is a multiset of traces, i.e. $L \subset \mathbb{B}(\mathcal{E}^*)$.
Predictive monitoring aims to estimate the future KPI values of the running cases. Here, we aim to be generic, meaning that KPIs can be of any nature:

[1] Given a set X, $\mathbb{B}(X)$ indicates the set of all multisets with the elements in X.

Definition 2.3 (KPI): Let $\mathcal{E}$ be the universe of events defined over a set $\mathcal{A}$ of attributes. Let $\mathcal{W}_K$ be the domain of the KPI values. A KPI is a function $\mathcal{T} : \mathcal{E}^* \times \mathbb{N} \not\rightarrow \mathcal{W}_K$ such that, given a trace $\sigma \in \mathcal{E}^*$ and an integer index $i \leq |\sigma|$, $\mathcal{T}(\sigma, i)$ returns the KPI value of σ after the occurrence of the first i events.[2]

Note that our KPI definition assumes it to be computed a posteriori, when the execution is completed and leaves a complete trail as a certain trace σ. In many cases, the KPI value is updated after each activity execution, which is recorded as next event in trace; however, other times, this is only known after the completion. We aim to be generic and account for all relevant cases. Given a trace $\sigma = \langle e_1, \ldots, e_n \rangle$ that records a complete process execution, the following are three potential KPI definitions:

Remaining Time. $\mathcal{T}_{remaining}(\sigma, i)$ is equal to the difference between the timestamp of e_n and that of e_i.

Activity Occurrence. It measures whether a certain activity is going to eventually occur in the future, such as an activity *Open Loan* in a loan-application process. The corresponding KPI definition for the occurrence of an activity A is $\mathcal{T}_{occur\_A}(\sigma, i)$, which is equal to true if activity A occurs in $\langle e_{i+1}, \ldots, e_n \rangle$ and $i < n$; otherwise false.

Customer Satisfaction. This is a typical KPI for several service providers. Let us assume, without losing generality, to have a trace $\sigma = \langle e_1, \ldots, e_n \rangle$ where the satisfaction is known at the end, e.g. through a questionnaire. Assuming the satisfaction level is recorded with the last event - say $e_n(sat)$. Then, $\mathcal{T}_{cust\_satisf}(\sigma, i) = e_n(sat)$.

The following definition states the prediction problem:

Definition 2.4 (The Prediction Problem): Let L be an event log that records the execution of a given process, for which a KPI $\mathcal{T}$ is defined. Let $\sigma = \langle e_1, \ldots, e_k \rangle$ be the trace of a running case, which eventually will complete as $\sigma_T = \langle e_1, \ldots, e_k, e_{k+1} \ldots, e_n \rangle$. The prediction problem can be formulated as forecasting the value of $\mathcal{T}(\sigma_T, i)$ for all $k < i \leq n$.

As indicated in Section I, we aim to provide an explanation for the predictions. In particular, for each running case, we aim to return the set of attributes influencing its prediction the most, with the corresponding magnitude and the indication whether the attributes increase or decrease the predicted KPI's value.

In the light of the above, for each trace $\sigma = \langle e_1, \ldots, e_n \rangle$, the problem can be stated as finding a function $\mathcal{K}_{(\sigma,\mathcal{T})}$ such that, for all $a \in \mathcal{A}$, $v \in \mathcal{W}_{\mathcal{A}}(a)$, and for all i s.t. $-n < i \leq 0$, $\mathcal{K}_{(\sigma,\mathcal{T})}(a, v, i)$ is different from zero if and only if the assignment of value v to attribute a by $e_{(n-i)}$ has influenced the prediction of KPI $\mathcal{T}$. The absolute value of $\mathcal{K}_{(\sigma,\mathcal{T})}(a, v, i)$ indicates how much this influence is, where a zero value indicates no influence. If $\mathcal{K}_{(\sigma,\mathcal{T})}(a, v, i) \neq 0$, its positive or negative sign indicates whether the influence is towards increasing or decreasing the KPI value:

Definition 2.5 (The Prediction-Explanation Problem): Let L be an event log over a set $\mathcal{A}$ of attributes, with domains $\mathcal{W}_{\mathcal{A}}$. Let $\sigma = \langle e_1, \ldots, e_k \rangle$ be a running case with a KPI definition $\mathcal{T}$. Let be $\overline{\mathcal{W}} = \cup_{a \in \mathcal{A}} \mathcal{W}_{\mathcal{A}}(a)$. Explaining the prediction is the problem of computing a function $\mathcal{K}_{(\sigma,\mathcal{T})} : \mathcal{A} \times \overline{\mathcal{W}} \times [-k+1, 0] \rightarrow \mathbb{R}$, where $v \notin \mathcal{W}_{\mathcal{A}}(a) \Rightarrow \mathcal{K}_{(\sigma,\mathcal{T})}(a, v, i) = 0$.

III. Related Works

A. Prediction of Process-Related KPIs

The predictive-monitoring survey of Márquez et al. [9] reports on the large repertoire of techniques and tools that were developed to address this problem. However, the authors claim that *"little attention has been given to [...] explaining the prediction values to the users so that they can determine the best way to act upon"*, and that *"it is necessary to develop tools that help users to query these models in order to get information that is relevant for them"*. These are in fact the problems tackled in this paper, so as to ensure that the predictive-monitoring system is trusted, and thus used.

Predictive monitoring has been built on different machine and deep-learning techniques, and also on their ensemble [9]. Different research works have recently illustrated that the so-called Long Short-Term Memory networks (LSTMs) generally outperform other methods (see, e.g., [14], [23], [12]). Therefore, while our explanation framework is independent of the machine- or deep-learning technique that is employed, we operationalize it with LSTMs. Section IV provides further details on LSTMs, and details how they are employed for business-process predictive monitoring.

It was explained above that little research work has been conducted on explaining the outcome of process predictive monitoring. The most relevant work is by Rehse et al. [15], which also aims at providing a dashboard to process participants with predictions and their explanation. However, the paper does not provide sufficient details on the actual usage of the explainable-AI literature, and the very preliminary evaluation is based on one single artificial process that consists of a sequence of five activities. Breuker et al. also try to tackle the problem [3], but their attempt is not independent of the actual technique employed for predictions. Furthermore, their explanations are only based on the activity names, while the explanations can generally involve resources, time, and more (cf. the case studies reported in Section VI).

B. Explanation of Machine-Learning Models

Few approaches exist in the literature to explain machine learning models, arisen from the need to understand complex black-box algorithms like ensembles of Decision Trees and Deep Learning [16], [20], [22], [7].

The adoption of explanatory methods in industry is at an early stage; in [21] an approach of fake news detection grounded in explainability is introduced. A significant amount of work in literature is focused on healthcare applications. We highlight [8], an implementation of the Shapley Values in healthcare, where the explanatory method is used to prevent

[2] Given a sequence X, $|X|$ indicates the length of X. Notation $\not\rightarrow$ indicates that the function is partial.

hypoxaemia during surgery, and [10], where explainability is used for analysis of patience re-admittance.

The SHAP implementation of the Shapley values for Deep Learning has the strong theoretical foundation of the original game theory approach, with the advantage of providing offline explanations that are consistent with the online explanations. Moreover, SHAP avoids the problems in consistency seen in other explanatory approaches (e.g. the lack of robustness seen in the online surrogate models, as analysed in [1]). The framework proposed in this paper specializes the use of Shapley values to the problem of providing explanations for predictive analytics.

We also considered attention mechanisms [2] as an alternative. However, two limitations made us opt for Shapley values. First, attention mechanisms necessarily have to be integrated in a Neural Network architecture, while Shapley values can be applied to any Machine or Deep Learning algorithm. The second limitation is linked to the lack of consensus that attention weights are always correlated to feature importance. Jain et al. [17] find it *"at best, questionable – especially when a complex encoder is used, which may entangle inputs in the hidden space"*, Serrano et al. [18] state that *"attention weights often fail to identify the sets of representations most important to the model's final decision"*.

IV. The Use of LSTMs for Predictive Monitoring

As indicated in Section I, we implemented our framework by leveraging on LSTM models [6], a special type of Recurrent Neural Networks. LSTM models natively support the predictions where the independent variables are sequences of elements, and the literature has shown that they are among the most suitable methods for predictive business monitoring (cf. Section III-A).

The construction of LSTM models fall into the problem of supervised learning, which aims to learn the model from a training set, for which the value of the dependent variable is known. This set is composed by pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$ where $\mathcal{X}$ represents the independent variables with their values (also known as **features**), and $\mathcal{Y}$ is the value observed for the dependent variable (i.e. the value we aim to predict).

In the domain of LSTM learning, $\mathcal{X}$ consists of sequences of vectors with a certain number n of dimensions, i.e. $\mathcal{X} = (\mathbb{R}^n)^*$.[3] When LSTM is used for predictive business monitoring using KPI values in a domain $\mathcal{W}_K$ (cf. Definition 2.3), $\mathcal{Y}$ is $\mathcal{W}_K$.

With these preliminaries at hand, we built a process monitoring framework composed by two phases: off-line and on-line.

The off-line phase requires an event log L and a KPI definition $\overline{\mathcal{T}}$ as input. This enables creating the dataset for training and testing the LSTM model, which consists of pairs $(x, y) \in (\mathbb{R}^n)^* \times \mathcal{W}_K$. The input is, hence, a sequence of vectors; conversely, a trace is a sequence of events. Therefore, each event needs to be encoded as a vector, which is a problem largely studied: we use the same encoding as in [12]; this can be abstracted as an **event-to-vector encoding function** $\rho : \mathcal{E} \rightarrow \mathbb{R}^n$. In a nutshell, each numeric attribute a of event e becomes a different dimension of $\rho(e)$, which takes on value $e(a)$. Each boolean attribute a is also a different dimension, with either 0 or 1 depending whether $e(a)$ is false or true. Each literal attributes a is represented through the so-called *one-hot encoding*: one different dimension exists for each value $v \in \mathcal{W}_\mathcal{A}(a)$, and the dimension referring to value $e(a)$ takes on value 1, with the other dimensions be assigned value 0. Function ρ can also be overloaded to traces: $\rho(\langle e_1, \ldots, e_m \rangle) = \langle \rho(e_1), \ldots, \rho(e_m) \rangle$.

The dataset is created starting from each prefix σ' of each trace $\sigma \in L$: σ' will generate one item in the data set consisting of a pair $(x, y) \in (\mathbb{R}^n)^* \times \mathcal{W}_K$ where $x = \rho(\sigma')$ and $y = \overline{\mathcal{T}}(\sigma, |\sigma'|)$. The dataset is later divided in one larger part for training the LSTM model, and a smaller part for testing. The test part is used to evaluate the quality of the LSTM model, in terms of different metrics. Details of the proportions and the quality metrics employed are discussed in Section VI. The **LSTM-based process predictor** trained from a dataset $\mathcal{D} \subset (\mathbb{R}^n)^* \times \mathcal{W}_K$ can be abstracted as a function $\Phi_\mathcal{D} : {\mathbb{R}^n}^* \rightarrow \mathcal{W}_K$.

The on-line phase aim is to predict the KPI of interest for a set of running cases of the process, identified by a set L' of partial traces (i.e., a log). It relies on the LSTM-based process predictor $\Phi_\mathcal{D}$: for each $\sigma' \in L'$, the predicted KPI value is $\Phi_\mathcal{D}(\rho(\sigma'))$.

V. Explanation of Generic KPI Predictions

This section reports on the main contribution of this paper, namely using Shapley Values to explain the predictions of any predictive model.

Section V-A introduces the theory behind Shapley values, while Section V-B illustrates its application and adaptation for predictive process monitoring. Then, in Section V-C we provide the general picture and the two main types of explanations reported.

A. The Theory of Shapley Values

The Shapley Values [19] is a game theory approach to fairly distribute the payout among the players that have collaborated in a cooperative game. This theory can be adapted as an approach to explain a predictive model. The assumption is that the features from an instance correspond to the players, and the payout is the difference between the prediction made by the predictive model and the average prediction (later referred to as the *base value*). Intuitively, given a predicted instance, the Shapley Value of a feature expresses how much the feature value contributes to the model prediction [11]:

Definition 5.1 (Shapley Value): Let $X = \{x_1, \ldots, x_n\}$ be a set of features. The Shapley value for feature x_i is defined as:

$$\psi_i = \textstyle\sum_{S \subseteq \{x_1,\ldots,x_m\} \setminus \{x_i\}} \frac{|S|!(p-|S|-1)!}{p!} \left(val\left(S \cup \{x_i\}\right) - val(S) \right)$$

where $val(T)$ is the so-called payout for only using the set of feature values in $T \subset X$ in making the prediction.

[3] In literature, LSTMs are often trained on the basis of matrices. However, a sequence of m vectors in $\mathbb{R}^n$ can be seen, in fact, as a matrix in $\mathbb{R}^{n,m}$. We use here the dataset representation as vectors to simplify the formalization.

Intuitively, the formula in Definition 5.1 evaluates the effect of incorporating the feature value x_i into any possible subset of the feature values considered for prediction. In the equation, variable S runs over all possible subsets of feature values, the term $val\,(S \cup \{x_i\}) - val(S)$ corresponds to the marginal value of adding x_i in the prediction using only the set of feature values in S, and the term $\frac{|S|!(p-|S|-1)!}{p!}$ corresponds to all the possible permutations with subset size $|S|$, to weight different sets differently in the formula. This way, all possible subsets of attributes are considered, and the corresponding effect is used to compute the Shapley Value of x_i.

B. Explainable Predictions through Shapley Values

The starting point is a event-to-vector encoding function $\rho : \mathcal{E} \to \mathbb{R}^n$ that maps each event to a feature vector (cf. Section IV). Given an event e_i, $\rho(e_i) = [x_i^1, \ldots, x_i^n]$ where each feature x_i^j is associated with an event attribute a_i^j and, possibly, with a value v_i^j. We mentioned that, if an attribute a_i^j is categorical, we need to introduce as many features as its possible values (one-hot encoding). Namely, x_i^j is both associated with an attribute a_i^j, and with a value v_i^j. If the feature associated with attribute a_i^j and value v_i^j takes on value 1, then $e(a_i^j) = v_i^j$; otherwise, the value is 0. If an attribute a_i^j is conversely numerical, only one feature x_i^j exists with value $e(a_i^j)$. When applied for explainable predictive monitoring, the Shapley values of a trace $\sigma = \langle e_1, \ldots, e_m \rangle$ are computed over the features of the vector $\chi = [x_1^1, \ldots, x_1^n, \ldots, x_m^1, \ldots, x_m^n]$ where $\rho(e_i) = [x_i^1, \ldots, x_i^n]$ for $1 \leq i \leq m$.

When applying Definition 5.1 to all features of χ, the result is a vector of Shapley values $\Psi = [\psi_1^1, \ldots, \psi_1^n, \ldots, \psi_m^1, \ldots, \psi_m^n]$ associated to feature vector χ, and attributes $[a_1^1, \ldots, a_1^n, \ldots, a_m^1, \ldots, a_m^n]$. Any Shapley value ψ_i^j can be either positive or negative. A positive or negative value indicates that the feature contributes to increasing or decreasing the value, respectively.

This allows us to construct the explanations. The first step is to determine which features are relevant and at which timestep.[4] For this, we consider the average μ of the values in Ψ along with their standard deviation ξ. This allows to define an interval $I = [\mu - \delta\xi, \mu + \delta\xi]$ of Shapley values that are not considered to contribute significantly, where δ is a parameter set by the user. This reduces the number of features that are considered in the explanation, limiting its verbosity.

Let us consider each Shapley value $\psi_i^j \notin I$, associated with feature x_i^j and an event's attribute a_i^j.

If a_i^j is a numerical attribute, attribute a_i^j is the explanation itself, i.e. $\forall \overline{v} \in \mathcal{W}_{\mathcal{A}}(a).\ \mathcal{K}_{(\sigma,\mathcal{T})}(a_i^j, \overline{v}, i-m) = \psi_i^j$.

If a_i^j is a categorical attribute, x_i^j is a one-hot encoded feature, and it is also associated with a value v_i^j. If $x_i^j = 1$, the explanation obtained is that $a_i^j = v_i^j$ contributes to the KPI value: $\mathcal{K}_{(\sigma,\mathcal{T})}(a_i^j, v_i^j, i-m) = \psi_i^j$. Otherwise, $x_i^j = 0$, and the explanation is $a_i^j \neq v_i^j$, namely $\forall \overline{v} \in \mathcal{W}_{\mathcal{A}}(a) \setminus \{v_i^j\}$. $\mathcal{K}_{(\sigma,\mathcal{T})}(a_i^j, \overline{v}, i-m) = \psi_i^j$.

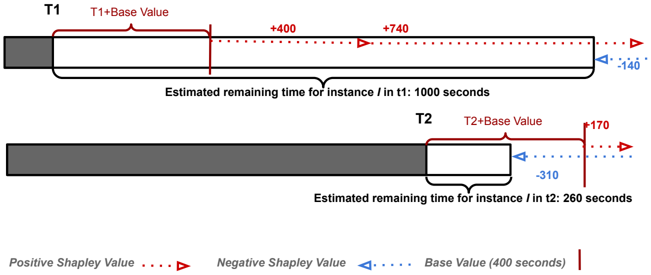
T1
T1+Base Value
+400
+740
-140
Estimated remaining time for instance *I* in t1: 1000 seconds
T2
T2+Base Value
+170
-310
Estimated remaining time for instance *I* in t2: 260 seconds
Positive Shapley Value
Negative Shapley Value
Base Value (400 seconds)

Fig. 1: Two explanations examples using Shapley Values. When the Remaining Time predicted is high (i.e. higher than Base Value), the Shapley Values indicate which features increase the prediction. Similarly, when the prediction is smaller than the Base Value, most of the Shapley Values are negative.

Any other combination (a, v, i) that does not fall into the situations above is such that $\mathcal{K}_{(\sigma,\mathcal{T})}(a, v, i) = 0$.

While an exact computation of the Shapley values requires to consider all combinations of features (hence, the algorithm is exponential on the number of features), efficient estimations can be obtained through polynomial algorithms that use greedy approaches [11].

To conclude, let us illustrate how Shapley values help explain a typical KPI in predictive process monitoring: estimated remaining time. Figure 1 shows the estimated remaining time of the same case in two different moments: T1, when the case started (upper figure, with an estimated remaining time of 1000 seconds), and T2, when it is close to its end (lower figure, with an estimated remaining time of 260 seconds). Considering that the Base Value is 400 seconds, the explanatory method would indicate, at T1, which features have been useful for the predictive model to predict a high value, i.e. the features with a positive Shapley Value. On the other hand, for T2, most of the Shapley Values would be negative, since the model has predicted a value smaller than the base value.

C. Overall Approach for Explaining Generic KPI Predictions

Explanations can be used offline to explain the features/factors that the trained model uses to make predictions, moreover they can be employed online on each running case to put forward the factors that affected the predictions. In particular, offline explanations are calculated on the test dataset, which is a part of the dataset not used for training the model (information about the division between train and test sets will be provided in Section VI).

*1) **Offline Explanations**:* Our offline explanation strategy is to provide an heatmap that overviews the importance of each factor in explaining the instances of the test dataset.

In particular, given an event log L, we consider each prefix σ' of each trace in L. Then, we compute the explanations as defined in Section V-B. Figure 2 shows an example of a heatmap reporting the frequency in which an explanation is relevant after each event in every trace. The y axis lists different explanations of types $attr = value$ or $attr \neq value$

[4] In this context each timestep refers to a different event of the trace along with its attributes (features). For instance, timestep zero refers to the first event of the trace, timestep one to the second, etc.

TABLE I: Online explanations for *Remaining Time* for three running cases. When the explanation is followed by (-1), it means that it refers to the value assigned to the attribute by the event that precedes the last of respective case.

CASE ID	REMAINING TIME	Explanations for increasing remaining_time	Explanations for decreasing remaining_time
201810011258	5d 6h 7m	ACTIVITY=Evaluating Request (NO registered letter)	CLOSURE_TYPE!=Inheritance
201810000206	5d 2h 12m	ROLE=DIRECTOR	CLOSURE_TYPE=Bank Recess
201811010829	2d 2h 31m	ROLE!=BACK-OFFICE (-1) AND ACTIVITY!=Service closure Request with BO responsibility (-1)	-
...	...	...	...

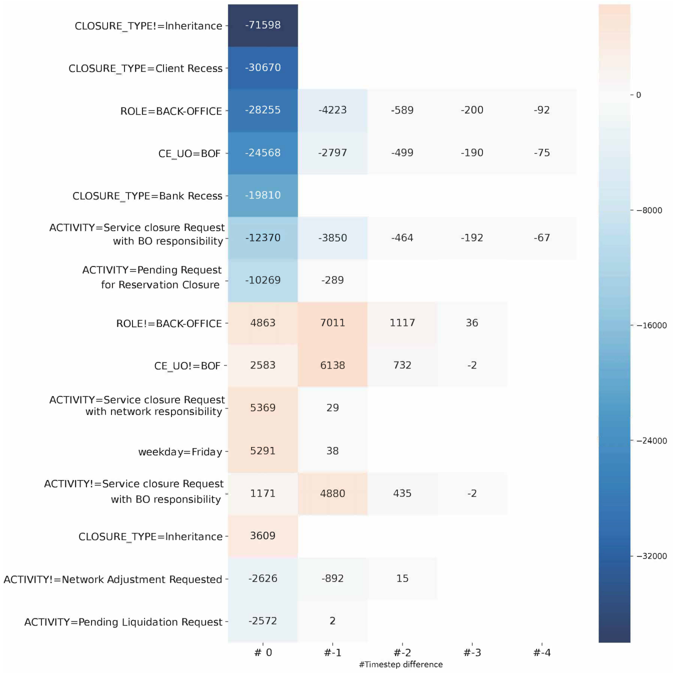

Fig. 2: The offline explanation of the remaining time

while the x axis lists the timestep difference between the event in question and the last event of the considered prefix, namely 0 indicates the last event, -1 indicates the second last, etc. A cell with explanation $a = v$ (y axis) and timestep difference t (x axis) takes on a value $(x - y)$ if there are x prefixes σ' of traces in L s.t. $\mathcal{K}_{(\sigma',\mathcal{T})}(a, v, t) > 0$ and y prefixes σ'' of traces in L s.t. $\mathcal{K}_{(\sigma'',\mathcal{T})}(a, v, t) < 0$. For instance, let us consider the explanation ROLE=BACK-OFFICE with timestep difference 0, which is associated with value -28255. This means that -28255 is the difference between the number of prefixes σ' in which ROLE=BACK-OFFICE in the last event of σ' contributes to increasing the KPI value and the number of prefixes σ' in which ROLE=BACK-OFFICE of last event contributes to decreasing. Similarly, -4223 is the difference when considering the second last event of the prefixes in place of the last. A similar reasoning can be repeated for explanations of type $a \neq v$. The heatmap uses different shades of blue and red to highlight the magnitude of negative and positive values, respectively.

*2) **Online Explanations**:* When we focus on running cases, we generate a table with one row per running case (see, e.g., Table I. Each row shows the case id, unique for each running case, the prediction for the current KPI, and the explanations that influence the prediction. Section VI discusses the case study in detail, including the results in Table I.

VI. Implementation and Experiments

The framework for explainable predictive monitoring has been implemented in Python, using Pandas to elaborate the data, and the shap library[5] to explain the prediction.[6] We relied on Keras framework for the LSTM implementation. The architecture was composed by 8 layers with 100 neurons each.

Each LSTM model was trained in 12-24 hours, and the computation of the off-line explanations (i.e. the heatmaps) required a similar amount of time. For each running case, on-line predictions and explanations are given in ca. half second. Note that training models in less than one day does not pose significant limitations: this is just performed once before putting the system in production.

The remainder of the paper will report on the experiments with different KPIs for the process carried on in an Italian bank. However, we also conducted several additional experiments with publicly-available event logs, which confirm the findings reported here. Space limitation prevent us from reporting on them, which are however discussed in the appendix of the extended version of this paper [5].

A. Domain description

Our assessment is based on the so-called ***Bank Account Closure***, a process executed at an Italian Banking Institution. The process deals with the closure of customer's accounts, which may be requested either by the customer or by the bank, for several reasons.

From the bank's information system, we extracted an event log with 32.429 completed traces and 212.721 events. It contains 15 different activities, 654 possible resources (recorded in an attribute labeled *Ce_Uo*), divided in 3 roles (attribute *role*). Each trace is associated with an attribute *Closure_Type*, which encodes the type of procedure that is carried out for the specific account holder, and the *Closure_Reason*, namely the reason triggering the closure's request. The latter is only known for 79.43% of cases.

For the bank, it is of interest to obtain an estimate of the remaining time until the end for running cases. This allows the bank to decide which cases require special attention, in order to not postpone them too much further. Also, the bank wants

[5] https://shap.readthedocs.io/en/latest

[6] Code can be found at https://github.com/PyRicky/LSTM_Generic_explainable

to be informed whether there are high chances that one or more of the following activities will occur: *Authorization Requested*, *Pending Request for Acquittance of heirs*, and *Back-Office Adjustment Requested*. They are linked to contingency actions, which should be avoided because they would cause inefficiencies in terms of time, costs, and resource utilization. Finally, the bank is also interested in obtaining an estimate of the total cost of a running case, in order to detect in advance which cases require particular attention.

We used two/thirds of the traces as training, and one third as test set. For improving the quality of the trained model, we used hyperparameter optimization, with 20% of the training data employed for this (validation set).

Sections VI-B, VI-C and VI-D report on the outcome for remaining-time prediction, for the prediction of the occurrence of one of those three contingency actions and for total cost prediction, respectively.

B. Results on Remaining Time

Section V showed that the explanation for a learnt prediction model is given as a heatmap during the offline phase. Figure 2 refers to the application for the remaining time prediction. The fact that the closure type is not Inheritance (*Closure_Type!=Inheritance*) is the largest value in the heatmap (as absolute value), so it is the largest factor that influences the prediction. The information that the value is negative (i.e. -71598) indicates that the influence is towards reducing the value, namely towards having lower remaining time. From a domain viewpoint, when the type of procedure is Inheritance, the bank-account holder is passed away. A further analysis of the data confirms this finding: if the type is *Inheritance*, the process duration is 29 days, versus 14 days when the type is different. The evidence in the explanation illustrates that LSTM allowed learning a prediction model that leverages on the closure type to estimate the remaining time. Other important attributes are related to the role associated to the resource and the resource performing each activity. Let us consider attributes *Role=Back-office* and *CE_UO=BOF* that are related to back-office activities, which are generally performed in the final part of cases; it can be seen in the heatmap that even in this case the model is able to predict that the process instance is about to complete (a negative value again indicates smaller remaining time).

The discussion was so far focused on the attribute of the last event. However, the values of attributes of previous events also influence the prediction of remaining time as shown in the heatmaps (see columns related to timestep differences -1, -2, -3 and -4). Consider, e.g., the row *ROLE=Back-office* and column -1: the value -4223 indicates that if the previous event refers to an activity performed by a resource with role Back-office, this influences to lower the prediction: the case is getting even closer to the end. When activities are performed by a resource director the behaviour is considered as exceptional, while activities performed by resources playing the role of applicant are in general performed in the initial part of cases; consequently, the cases usually take longer to complete. This is indicated by the positive value 4863 of the last event in the row *ROLE!=Back-office*, which indicates that the influence is towards increasing the remaining time. Notice that the column related to timestep difference -1 has a bigger value (7011), indicating that if the previous event refers to an exceptional activity, the influence on the prediction will be even stronger. Finally when the activity performed is other than Network Adjustment Requested then the predicted remaining time is smaller; this is in fact an exceptional activity, that only occurs when an error is made in the early stages of the process, and even in this case our framework was able to learn to predict a smaller remaining time when no adjustments need to be done.

Section V indicated that explanations are also given for running cases to explain predictions to process stakeholders. Our implementation returns a CSV file with the predictions for the running cases; a subset is provided in Table I, which shows the factors that increase or decrease the prediction for the remaining time prediction. Let us consider as an example the last row: the remaining time is predicted as being ca. 2 days and 2 hours, with two explanations increasing the prediction, one related to the fact that the previous activity performed was not *Service Closure Request with BO Responsibility*, and the other related to the resource performing the previous activity with a role not being *Back-Office*.

To conclude, since this KPI is numerical and the values are reasonably well balanced, we adopted the *Mean Absolute Error* (MAE), which is the average difference between the actual and the predicted value, computed over all test-set samples. Here, we achieved a MAE of 4.37 days, which is around the 28% of the average case duration (i.e. 15.5 days).

C. Results on Prediction of Activity Occurrence

We mentioned that the financial institute aims to avoid activities related to inefficiencies (e.g. rework): *Pending Request for Acquittance of Heirs*, *Back-Office Adjust Requested* and *Autorization Required*. Space limitation prevents us from showing here all of three: here we focus on activity *Back-Office Adjust Requested*, while the other two are in the appendix complementing the paper [5]. The learnt LSTM model was characterized by an F1 score of 0.65, an Area Under the Receiver Operating Charateristics (AUROC) of 0.86, and an Area under Precision/Recall curve (APR) of 0.69. We computed AUROC and APR, because these metrics are, in fact, more suitable when some classes are unbalanced. This is actually the case for our case study: the three activities are contingency actions, which occur infrequently.

The heatmap related to *Back-Office Adjustment Requested* prediction (Figure 3) shows that the attributes related to the type and the reason of bank account closure are influencing the most. When all bank accounts of a customer are closed (labeled by *Closure_Reason=1 - Client lost*) or when the customer decides to close one of its bank accounts among different ones he owns (labeled as *Closure_Reason=2 - Keep bank account. Same dip*), then a *Back-Office Adjustment Requested* is unlikely to happen. This is clearly shown in the heatmap, respectively represented by the values -40374

TABLE II: Online explanations for *Back-Office Adjustment Requested*. Values 1 and 0 indicate if the activity is predicted to occur or not. Explanation followed by (-1): attribute value assigned by the event that precedes the last of respective case.

CASE ID	**Back-Office Adjustment Requested**	**Explanations for Back-Office Adjustment Requested happening**	**Explanations for Back-Office Adjustment Requested not happening**
201810000206	0	-	ACTIVITY=Service closure Request with network responsibility (-2) AND CE_UO=195 (-1)
201811008237	1	CLOSURE_TYPE=Porting	-
201812005701	1	CLOSURE_REASON!=1 - Client lost	-
...	...	...	...

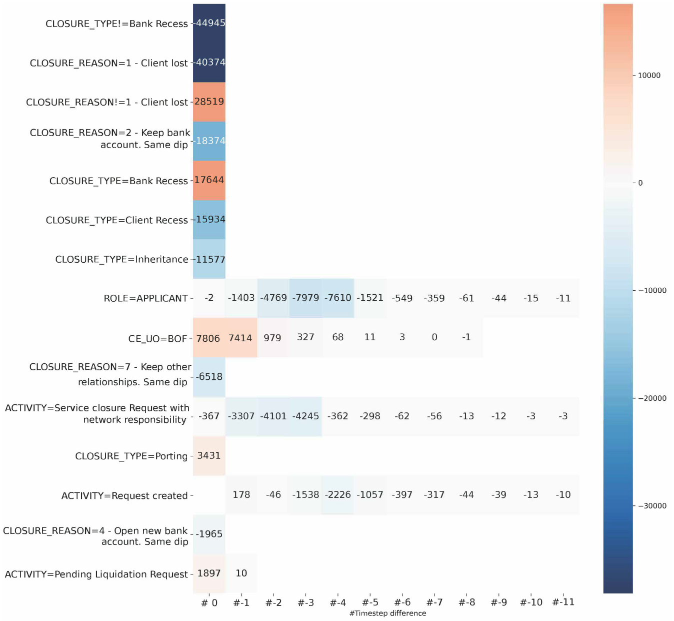

Fig. 3: Offline explanations for *Back-Office Adjustment Requested*

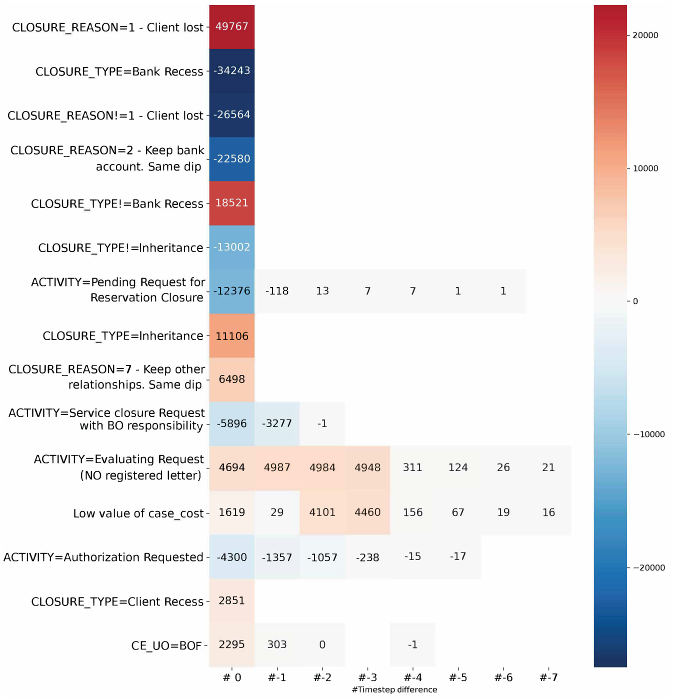

Fig. 4: Offline explanations for *Case cost*

and -18374, which influence is towards not predicting the occurrence of this activity. Values -15934 when the Closure Type is Client Recess (it is the client that decides to close the bank account) and -11577 when it is Inheritance (the bank-account holder is passed away) indicate as well that a *Back-Office Adjustment Requested* is unlikely to happen. Conversely, when the Closure Type is Bank Recess (the bank account is closed by the bank) or it is Porting, then the rework activity *Back-Office Adjustment Requested* is more likely to occur.

Explanations are also used on-line to explain the predictions of running cases. Table II shows the factors that make the model predict whether or not activity *Back-Office Adjustment Requested* is expected to happen for three running cases. Values 1 and 0 indicate that the activity is expected or not to happen, respectively. Let us consider for instance the first case in the table: the rework activity is not expected to happen because two events ago *Service Closure Request with Network Responsibility* has been performed and because the previous event has been performed by the resource 195. Conversely, it is predicted to eventually happen for the other two cases in the table, and the explanation is related to the closure type being Porting and the closure reason not being Client lost.

D. Results on Case Cost prediction

Since this KPI is numerical, we adopted the *Mean Absolute Error*, for which we achieved a value of 0.95 Euros. This is an excellent result, given that the average case cost is 12.86, with standard deviation of 6.41. Figure 4 shows the application for the case cost prediction for the off-line phase. The main factor that contributes to increase the cost of a case is represented by *Closure_Reason=1 - Client Lost*, which is indicated when all bank accounts are going to be closed. The information that the value is positive (i.e. 49767) indicates that the influence is towards increasing the cost. This is mainly caused by the fact that most of the times here the director needs to carefully evaluate the request before proceeding, and the hourly director's wage is certainly higher than that of other bank employees. Nevertheless, this evaluation is not needed when the closure of the bank account is requested by the bank (labeled as *Bank Recess*), therefore the predicted case cost will be smaller (indicated in the heatmap by the negative value -34243). The director is similarly not involved when customers only close one of their bank accounts (*Closure_Reason=2 - Keep bank* *account. Same dip*), which is a factor that yields lower costs. Another reason is that when only one between different bank accounts is closed, then of course the process is simpler and less Back-office adjustment activities need to be performed compared to when all bank accounts need to be closed, leading to minor costs. Another indirect evidence that the director's involvement is a factor that increases costs is evident when one looks at the explanations based on *Activity=Evaluating Request (NO registered letter)*. This activity needs a lot of time and is performed by the director, leading to high costs (even higher compared to the case in which a request has only to be authorized). If this activity occurs, the cost will remain permanently high. This is evident in the heatmaps: the fact that this activity has been previously performed is still influencing towards increasing the costs (see columns related to timestep difference -1, -2 and -3, which values are respectively 4987, 4984 and 4948).

VII. Conclusion

A lot of research has been devoted towards increasingly accurate frameworks for predictive process monitoring. Nonetheless, little attention has been paid to ensure that that the resulting predictive-monitoring system is workable in practice. With practical workability, here we intend that the process analysts and stakeholders need to trust the system and its predictions. Previous studies have shown that a necessary condition to build trust is to explain the reason of the provided predictions [13], [4]. Proposals that do not put explanation as a core feature are not going to be adopted in practice.

This paper has put forward a framework to equip predictive-process-monitoring systems with explanations that are intelligible by actors of the process. The framework builds on the most recent state of the art on Explainable AI, and is independent of the actual AI predictive-analytics technique.

However, the operationalization of the framework requires one to select an actual AI technique, and here we opted for predictive models based on LSTM, which the present literature has shown to be the most suitable for the problem in question. The implementation is based on Python, and it has been used for several case studies. Here we reported different KPI predictions for a process run in a financial institute in Italy. The case studies shows that our framework is able to, on the one hand, provide explanations of the most salient features that influence the prediction models and, on the other hand, to provide online explanations on the running cases.

Future work accounts different directions. First, we aim to verify through interviews whether process stakeholders would fully comprehend the heatmaps and the form given to explanations. Second, we aim to explore the possibilities of *Natural Language Generation* techniques to report more user-friendly explanations, instead of the output shown in Tables I and II.

Acknowledgement. This work is supported by MINECO and FEDER funds under grant TIN2017-86727-C2-1-R and by the Department of Mathematics, University of Padua, with HPC resources.

References

[1] Alvarez-Melis, D., Jaakkola, T.S.: On the robustness of interpretability methods. arXiv preprint arXiv:1806.08049 (2018)

[2] Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: The 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)

[3] Breuker, D., Delfmann, P., Matzner, M., Becker, J.: Designing and evaluating an interpretable predictive modeling technique for business processes. In: Business Process Management Workshops. pp. 541–553. Springer (2015)

[4] Doshi-Velez, F., Kim, B.: Towards a rigorous science of interpretable machine learning (2017)

[5] Galanti, R., Coma-Puig, B., de Leoni, M., Carmona, J., Navarin, N.: Explainable predictive process monitoring. arXiv:2008.01807 (2020)

[6] Hochreiter, S., Urgen Schmidhuber, J.: Long Short-Term Memory. Neural Computation **9**(8), 1735–1780 (1997)

[7] Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Advances in neural information processing systems. pp. 4765–4774 (2017)

[8] Lundberg, S.M., Nair, B., Vavilala, M.S., Horibe, M., Eisses, M.J., Adams, T., Liston, D.E., Low, D.K.W., Newman, S.F., Kim, J., et al.: Explainable machine-learning predictions for the prevention of hypoxaemia during surgery. Nature biomedical engineering **2**(10), 749 (2018)

[9] Márquez-Chamorro, A.E., Resinas, M., Ruiz-Cortés, A.: Predictive monitoring of business processes: A survey. IEEE Transaction on Services Computing **11**(6), 962–977 (2018)

[10] Meacham, S., Isaac, G., Nauck, D., Virginas, B.: Towards Explainable AI: Design and Development for Explanation of Machine Learning Predictions for a Patient Readmittance Medical Application, pp. 939–955 (06 2019)

[11] Molnar, C.: Interpretable Machine Learning (2020)

[12] Navarin, N., Vincenzi, B., Polato, M., Sperduti, A.: LSTM networks for data-aware remaining time prediction of business process instances. In: Proceedings of the IEEE Symposium Series on Computational Intelligence (SSCI 2017) (2017)

[13] Nunes, I., Jannach, D.: A systematic review and taxonomy of explanations in decision support and recommender systems. User Modeling and User-Adapted Interaction **27**(3–5), 393–444 (Dec 2017)

[14] Park, G., Song, M.: Prediction-based resource allocation using lstm and minimum cost and maximum flow algorithm. In: International Conference on Process Mining (ICPM). pp. 121–128 (2019)

[15] Rehse, J.R., Mehdiyev, N., Fettke, P.: Towards explainable process predictions for industry 4.0 in the dfki-smart-lego-factory. KI - Künstliche Intelligenz **33**(2), 181–187 (Jun 2019)

[16] Ribeiro, M.T., Singh, S., Guestrin, C.: "why should I trust you": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco. pp. 1135–1144 (2016)

[17] Sarthak, J., Wallace, B.C.: Attention is not explanation. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 3543–3556. Association for Computational Linguistics (2019)

[18] Serrano, S., Smith, N.A.: Is attention interpretable? In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 2931–2951. Association for Computational Linguistics, Florence, Italy (2019)

[19] Shapley, L.S.: A value for n-person games. Contributions to the Theory of Games **2**(28), 307–317 (1953)

[20] Shrikumar, A., Greenside, P., Kundaje, A.: Learning important features through propagating activation differences. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 3145–3153. JMLR. org (2017)

[21] Shu, K., Cui, L., Wang, S., Lee, D., Liu, H.: Defend: Explainable fake news detection. In: International Conference on Knowledge Discovery & Data Mining, SIGKDD. pp. 395–405. ACM (2019)

[22] Sundararajan, M., Taly, A., Yan, Q.: Axiomatic attribution for deep networks. In: Proceedings of the 34th International Conference on Machine Learning-Volume 70. pp. 3319–3328. JMLR. org (2017)

[23] Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Proceedings of 29th International Conference on Advanced Information Systems Engineering (CAiSE 2017). pp. 477–492 (2017)