Model-informed LIME Extension for Business Process Explainability

Guy Amit[1,†], Fabiana Fournier[1,†], Shlomit Gur[1,†] and Lior Limonad[1,†]

[1]IBM Research - Haifa, Israel

Abstract

Our focus in this work is on the adaptation of eXplainable AI techniques for the interpretation of business process execution results. Such adaptation is required since conventional employment of such techniques involves a surrogate machine learning model that is trained on historical process execution logs. However, being a data-driven surrogate, its representation faithfulness of the real business process model affects the adequacy of the explanations derived from it. Hence, native use of such techniques is not ensured to be adhering to the target business process explained. We present a business-process-model-driven approach that extends LIME, a conventional machine-learning-model-agnostic eXplainable AI tool, to cope with business processes constraints that is replicable and reproducible. Our results show that our extended LIME approach produces correct and significantly more adequate explanations than the ones given by LIME as-is.

Keywords

eXplainable Artificial Intelligence, Business Process, Augmented Business Process Management System, Machine Learning, Situation-Aware eXplainability

1. Introduction

The wide penetration of Artificial Intelligence (AI) into Business Process Management Systems (BPMSs) brings along many benefits and is considered by many as "the next disruptive technology that will touch almost all business process activities performed by humans." [1, 2]. However, the suspiciousness of human operators in the presence of AI comes with adoption hesitance, especially when the focus of the automation involves the replacement of human labor. Contemporary studies show that eXplainable Artificial Intelligence (XAI) is one of the major factors for user adoption of AI-based systems. For example, the "Global AI Adoption Index 2021" recent study [3], conducted by Morning Consult, denotes that more than 90 percent of companies using AI say their ability to explain how it arrived at a decision is critical. The need to be able to provide explanations is also reinforced by recent regulations. or example, the General Data Protection Regulation (GDPR) [4] in Europe enforces "the right [...] to obtain an explanation of the decision reached" and the European Commission recently published an ethics guideline for trustworthy AI (High-Level Expert Group on AI, 2019), where the need to explain

PMAI@IJCAI22: International IJCAI Workshop on Process Management in the AI era, July 23, 2022, Vienna, Austria

[†]These authors contributed equally.

guy.amit@ibm.com (G. Amit); fabiana@il.ibm.com (F. Fournier); shlomit.gur@ibm.com (S. Gur); liorli@il.ibm.com (L. Limonad)

0000-0001-6569-1023 (F. Fournier); 0000-0001-5174-3689 (S. Gur); 0000-0002-4784-2147 (L. Limonad)

a model's decisions is deemed an essential requirement for establishing trust between human users and AI systems. Similar legislation is also established in the US, such as the California Consumer Privacy Act (CCPA) of 2020 [5].

In their position paper [6], 14 researchers from nine different institutions laid the groundwork for a new generation of BPMSs that are augmented with AI, which they coined "Augmented Business Process Management Systems (ABPMSs)". The goal of these ABPMSs is to enhance the execution of traditional business processes with novel AI-based capabilities, making them more adaptable, proactive, explainable, and context-sensitive. One of the main characteristics of ABPMSs is their ability to explain and reason about processes executions. This is vital to enable the people who engage with ABPMS to be convinced that the system they operate can be relied upon and act accordingly. However, finding an adequate explanation is not easy, because this requires understanding the situational conditions in which specific decision were made during process enactments. Frequently, explanations cannot be derived from "local" inference (e.g., current undergoing task or decision in a business process) but require reasoning and understanding of situation-wide contextual conditions, relevant to the current step derived from some actions in the past. The authors of this manifesto identify "Situation-Aware eXplainability (SAX)" as one of the most prominent research challenges that ABPMSs call for. SAX aims at being able to entail ongoing tracking of inferential associations between subsequent enactments as a basis to gaining confidence in its ability to provide trustworthy explanations.

Previous work regarding explainability in the context of business processes calls for "process-aware explainability". Jan et al. [3] relate to explainability as being local to the context in which AI operates along the overall (global) reasoning process and motivate the need for process-aware explanations. This is reinforced in a more recent paper [7] by showing that native use of an XAI tool such as Local Interpretable Model-agnostic Explanations (LIME) [8] may drive misleading and even wrong explanations as it does not take into account BPs' constraints. Our proposed approach extends their work by applying a BP-model-driven approach, providing a comprehensive methodology that extends LIME to cope with BPs' constraints. Moreover, it can be replicable and reproducible in ABPMSs as a first step towards full SAX BPMSs.

2. Background

The use of ML models has become more wide spread in recent years [9, 10, 11]. Applications include domains that could potentially greatly affect individuals as well as society as a whole, such as medical diagnostics, credit scoring, recidivism prediction, and autonomous driving. At the same time, advancements in the ML field have increased the complexity of ML models, often at the expense of explainability, leading to "black box" ML models. Therefore, we have also seen an increase in the need to explain ML models and their predictions, fueled in part by legislation, but also by incentives from the user's or stakeholder's point of view (e.g., justify the ML model and gain domain insight), and from the developer's point of view (e.g., evaluate and improve the ML model). Although some ML models are inherently explainable (e.g., decision trees and linear models) and their internal logic predictions are interpretable (i.e., can be easily understood by humans), more complex models (e.g., ensemble models and deep learning models) require external explanation frameworks, namely XAI, in order to be human-understandable [9, 10].

XAI frameworks are predominantly post-hoc [9, 10]; that is, they are applied to the ML model after its training has been completed. Context-wise they can be divided into global, local, and hybrid explanations [10, 12, 13]. Global explanations attempt to explain the ML model's internal logic, local explanations try to explain the ML model's prediction for a single input instance, and hybrid approaches vary (e.g., explaining the ML model's internal logic for a subspace of the input space).

Our main focus in this work is on the adaptation of ML-model-agnostic post-hoc local XAI frameworks that are compatible with tabular data for the interpretation of BP execution results. Such adaptation is needed since conventional employment of XAI comes with the need to assess a variety of model inputs against its resulting outputs via model replays. Such replays are not pragmatic using the real-world BP model, as the execution of each case takes a long time, may involve human decision-making that applies tacit knowledge, and may cost money. Hence, the use of XAI requires the development of a surrogate ML model (e.g., a decision-tree) that represents the real one. These surrogate ML models are typically trained using historical process execution logs. Therefore, being a data-driven surrogate, its faithfulness to the real BP model may be lacking, which in turn affects the adequacy of the explanations derived from it. In this work, we show how additional knowledge that can be derived from the original BP model promotes the adequacy of such explanations, while employing a conventional ML-model-agnostic XAI tool.

The most commonly used ML-model-agnostic post-hoc local XAI frameworks, compatible with tabular data, are LIME [8] and SHapley Additive exPlanations (SHAP) [14]. LIME is based on the interpretability of linear models, whereas SHAP borrows from the domain of Game Theory and is based on the concept of Shapely additive values. Although different, the two share one important concept - both frameworks rely on a process of sampling data points (by way of feature perturbations), thereby providing a local neighborhood around the examined sample. Similar to [7], we focus on LIME, because its code is easier to modify than the code of SHAP. In addition, data-driven inter-feature dependencies are not considered in LIME, which is a benefit of using it over SHAP, since in our case, we prefer to have such dependencies inferred from the BP model rather than be [partially] elicited from the dataset.

2.1. LIME: Local Interpretable Model-agnostic Explanation

LIME can be used to generate both local and global explanations; in this work we focus on the local explanation. LIME works under the assumption that within a local neighborhood of a sample instance, the ML model's predictions depend on a subset of features. This assumption causes the ML model to provide explanations that are locally-faithful. That is, features that are important in the prediction on one instance may not be important in the prediction on another or in the global scope of the data set.

LIME explanation process begins by generating a sample neighborhood, which is a set of k data points $\{\bar{x}_i\}_{i=1}^{k}$, around an instance x. Numeric features are sampled using a continues distribution (e.g., Gaussian) around a feature value, which can be either the value of an instance to be explained (local neighborhood) or the global expected value of the feature, based on the training set (global neighborhood). Categorical features are sampled according to each category's frequencies in the training data. It is important to note that the inter-feature dependencies are

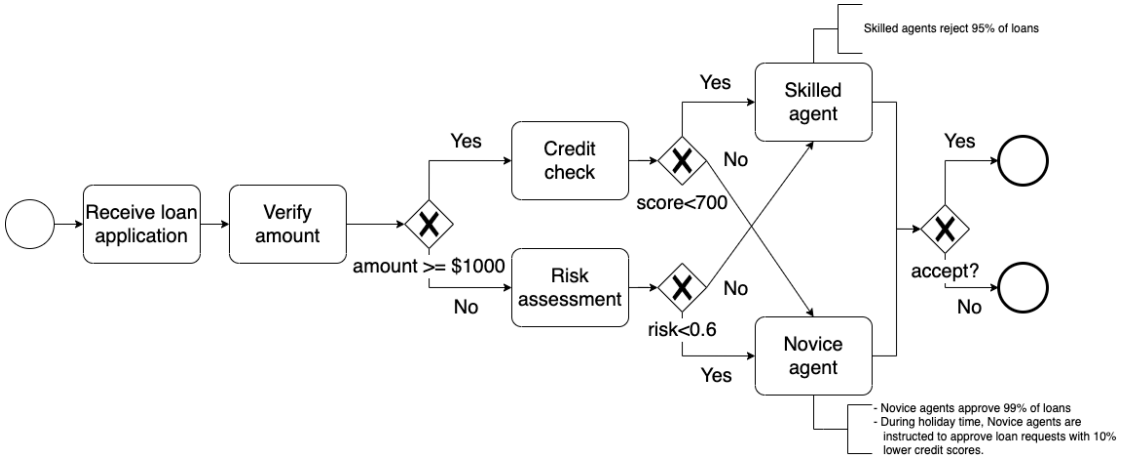

Figure 1: BP model in BPMN of the loan application process.

not considered in this sampling approach, which is a critical limitation of LIME from a BP point of view.

After the sampling process, the ML model to be explained, $\mathcal{M}(\cdot)$, is used to provide predictions: $\bar{y}_i = \mathcal{M}(\bar{x}_i)$. The resulting labeled data set $\{\bar{x}_i, \bar{y}_i\}_{i=1}^{k}$ is then used to train a sparse linear model $f(\cdot)$ with a weighted loss function, which can be formulated as:

$$\mathcal{L} = \sum_{i=1}^{k} \mathcal{D}(x, \bar{x}_i) \left(f(\bar{x}_i) - \bar{y}_i\right)^2 + \Omega(f)$$

Where $\mathcal{D}(\cdot, \cdot)$ is a distance function and $\Omega(\cdot)$ is a function ensuring the sparsity of the model's weights. The weights of the resulting $f(\cdot)$ reflect the behavior of $\mathcal{M}(\cdot)$ within the sample neighborhood and therefore serve as a local feature importance metric.

3. Illustrative Example

Figure 1 depicts a loan application BP in which each loan application goes through a set of predefined set of tasks (e.g., verify amount and credit check) and decision gateways (e.g., amount >= 1000) resulting in either the acceptance or the rejection of the loan application. We follow the Business Process Management Notation (BPMN) [1] to model the loan application. In our work we assume that all traces are complete. That is, we have a complete sequence of the BP from start to finish and we do not have partial traces or "in-flight" process executions.

Given the BP in figure 1, sets of rules are derived from it systematically as listed below. We denote each task execution in the BP model by a corresponding Boolean variable, named as the task's capitalised name, that is by default initialised with a $false$ value and gets assigned a $true$ value upon task execution. Internal BP variables are variables associated with conditional splits and are denoted by a lowercase naming convention.

[1] https://www.bpmn.org/

1. Rules derived from the gateways:

   a) $amount \geq 1000 \ \& \ Credit\_check$
   b) $amount < 1000 \ \& \ Risk\_assessment$
   c) $risk < 0.6 \ \& \ Novice\_agent$
   d) $risk \geq 0.6 \ \& \ Skilled\_agent$
   e) $credit\ score < 700 \ \& \ Skilled\_agent$
   f) $credit\ score \geq 700 \ \& \ Novice\_agent$

2. Activity indicators:

   a) $Verify\_amount \ \& \ amount \neq \text{NaN}$
   b) $Credit\_check \ \& \ credit\ score \neq \text{NaN}$
   c) $Risk\_assessment \ \& \ risk \neq \text{NaN}$
   d) $credit\ score \oplus risk$
   e) $Skilled\_agent \ \& \ accept \neq \text{NaN}$
   f) $Novice\_agent \ \& \ accept \neq \text{NaN}$

3. Entailment (backtracking) rules:
   a) $[accept \neq \text{NaN}] \models [Skilled\_agent \parallel Novice\_agent]$
   b) $Skilled\_agent \models [Credit\_check \parallel Risk\_assessment]$
   c) $Novice\_agent \models [Credit\_check \parallel Risk\_assessment]$
   d) $Credit\_check \models Verify\_amount$
   e) $Risk\_assessment \models Verify\_amount$
   f) $Verify\_amount \models Review\_loan\_application$
4. The note regarding instruction to novice agents during holiday time: Approve all loan applications with credit score that is 10% lower than the usual threshold.
5. The note regarding the acceptance rates of the agents implies that:
   a) $\frac{\left|\left\{x \in X | (x_{Skilled\_agent} = true) \cap (x_{accept} = true)\right\}\right|}{\left|\left\{x \in X | x_{Skilled\_agent} = True\right\}\right|} = 0.95$
   b) $\frac{\left|\left\{x \in X | (x_{Skilled\_agent} = false) \cap (x_{accept} = false)\right\}\right|}{\left|\left\{x \in X | x_{Skilled\_agent} = false\right\}\right|} = 0.01$

   Where X denotes the collective set of reviewed application traces

It is worth noting that in this work we only consider the basic sequencing of activities as expressed in the 'entailment' segment of the rules, where we acknowledge that such requirements could be further extended to express more complicated timing constraints associated with arrow semantics to be adhered to.

4. Methodology

In this section we elaborate on the steps to be executed in order to systematically augment the LIME Tabular Explainer to respect the constraints imposed by the BP model.

While LIME [8] is an open-source framework, it does not directly expose the feature perturbation process to its users. To test the effect of manipulating the neighborhood created by LIME perturbations, we used in-house code that modified a particular step in the LIME Tabular Explainer's process. We let the explainer generate a neighborhood, which we then exported to (1) edit and prune the neighborhood (see sections 4.2.1 and 4.2.2, respectively), (2) only prune it, or (3) leave it as-is. Next, we imported back into the explainer the resulting

neighborhood and let LIME continue its process as-is, computing local feature importance based on this neighborhood.

Variation (3) (keeping the neighborhood as-is) was introduced to guarantee a fair comparison to LIME without BPM-awareness. The results (explanations) for this variation were comparable to the results using a LIME Tabular Explainer without exporting and importing the neighborhood (results omitted). Our code is available at: https://github.com/IBM/SAX

4.1. Handling Missing Values

When encoding a BP data set, some values may be missing (e.g., if the amount in the loan application is less than $1000, then there is no credit score). However, many ML models, including SKLearn's [15] decision tree classifier, which we used in the current work (with default parameters), cannot handle missing values. Additionally, in the unmodified LIME Tabular Explainer, the distribution parameters (e.g., mean and standard deviation for Gaussian distributions) per feature are computed based on the training data and cannot handle missing values either. Therefore, as a first step, all missing values were replaced with -1's in the data, thereby affecting the perceived distribution of certain features. To eliminate this undesired effect, we modified the extended version of LIME to compute the distribution parameters based only on valid values (which in our case were values greater than or equal to 0).

4.2. Translation of Business Process Rules to Code

Our code implementation employs a combination of two strategies: editing and pruning (see sections 4.2.1 and 4.2.2, respectively). The rules in our illustrative example were translated to code as follows:

1. Rules derived from gateways were used in both editing and pruning as-is.
2. The activity indicators are inherent in the data representation (under the assumption of complete and error-free traces), with the exception of rules *b*, *c* and *d*, which were used in both editing and pruning.
3. The entailment rules are also inherent in the data representation, with the exception of rules *b* and *c*, which were used only in pruning, as they do not provide actionable rules regarding which of the two options should have taken place (a credit check or a risk assessment).
4. Rules that may cater to situational conditions with attributes that are not explicit in the BP model (e.g., *Is_holiday*) are not considered in the scope of the current methodology, and therefore were omitted from the translation to code. We plan to accommodate such "contextual annotations" rules in future work.
5. The rules pertaining to the acceptance rates apply to the global population, but may not hold in local neighborhoods within the population, and therefore were not used in the editing or in the pruning.

4.2.1. Editing the Neighbors According to the Business Process Rules

As previously mentioned, LIME's perturbations may result in instances that do not adhere to the BP rules. Therefore, we edited the neighborhood instances produced by LIME in accordance

with the process flow in figure 1 and to match the reference sample's BP model route. As a first processing step, we replaced all features that have NaN placeholder in the reference instance to NaN in all the neighbors, thereby forcing them to belong to the same BP model route. Next, editing may apply any of the two operators: (1) toggling value of a Boolean feature and (2) assigning NaN placeholder to a numeric feature.

In our illustrative example, after the NaN correction we enforced the BP rules *1.a* and *1.b*, then *2.b-2.d*, and then either *1.c-1.d* or *1.e-1.f*.

4.2.2. Pruning the Neighborhood According to the Business Process Rules

In order to create a BP-adhering neighborhood, instances that do not conform to the BP rules should be removed. Following the editing step, certain instances may still not conform to some of the BP rules (e.g., have neither *credit_score* nor *risk*). Therefore pruning was performed after editing.

In our illustrative example, we checked the instances against BP rules *1.a-1.f*, *2.b-2.d*, and *3.b-3.c*. If any of these BP rules were violated, the instance was excluded from the local neighborhood that was later used by LIME for the explanation.

5. Evaluation

5.1. Synthetic Data

Overall, our raw data set's structure conformed to the tuple: $t = \langle x_{amount}, x_{credit\_score}, x_{risk}, x_{Done\_credit\_check}, x_{Done\_risk\_assessment}, x_{Done\_skilled\_agent}, x_{Done\_novice\_agent}, x_{Done\_accept} \rangle$

In accordance with figure 1, we populated the raw data set (X) with 1000 instances of type t ($|X| = 1000$) with features drawn at random from Gaussian distributions. $\forall x \in X$, $x_{amount} \sim \mathcal{N}(1000, 200)$, $x_{Credit\_score} \sim \mathcal{N}(700, 200)$, and $x_{risk} \sim \mathcal{N}(0.6, 0.15)$. $\forall x \in X$, if $x_{amount} \geq 1000$, we set $x_{risk} = \text{NaN}$, $x_{Done\_credit\_check} = true$, and $x_{Done\_risk\_assessment} = false$; otherwise we set $x_{credit\_score} = \text{NaN}$, $x_{Done\_credit\_check} = false$, and $x_{Done\_risk\_assessment} = true$.

Next, $\forall x \in X$, if $x_{credit\ score} \geq 700$, we set $x_{Done\_skilled\_agent} = false$ and $x_{Done\_novice\_agent} = true$; otherwise, we set $x_{Done\_skilled\_agent} = true$ and $x_{Done\_novice\_agent} = false$. Similarly, $\forall x \in X$, if $x_{risk} \geq 0.6$, we set $x_{Done\_skilled\_agent} = true$ and $x_{Done\_novice\_agent} = false$; otherwise, we set $x_{Done\_skilled\_agent} = false$ and $x_{Done\_novice\_agent} = true$.

To remove redundancy in the representation, we unified mirroring features: $\forall x \in X$, $x_{Is\_credit} = x_{Done\_credit\_check}$ and $x_{Is\_skilled} = x_{Done\_skilled\_agent}$ and then we removed the features $Done\_risk\_assessment$, $Done\_credit\_check$, $Done\_skilled\_agent$, and $Done\_novice\_agent$.

Finally, to implement the rejection rates of the two agent types (95% for skilled agent and 1% for novice agent), we normalized the credit and risk scores to make them comparable. For details, refer to code in github (see in 4). Output example of this process can be seen in Table 1.

Table 1
Example BP records. '*amount*' indicates the amount of the loan request. '*credit_score*' is the applicant's credit score in case a credit check was performed (see '*Is_credit*') and NaN otherwise. '*risk*' is the applicant's risk score in case a risk assessment was performed (see '*Is_credit*') and NaN otherwise. '*Is_credit*' indicates whether the BP included a credit check (*true*) or a risk assessment (*false*). '*Is_skilled*' indicates whether the BP was processed by a skilled agent (*true*) or a novice agent (*flase*). '*accept*' indicates whether the loan request was approved (*true*) or rejected (*false*). The explanations for the first four records correspond to *a*, *b*, *c*, and *d* in Figure 2, in order.

amount	*credit_score*	*risk*	*Is_credit*	*Is_skilled*	*accept*
\$1024.7716	422.4298	NaN	true	true	false
\$927.0857	NaN	0.40616	false	false	true
\$1245.4166	520.2818	NaN	true	true	false
\$1290.8104	528.5517	NaN	true	true	false

5.2. Evaluation Criteria

After setting a random seed (for reproducibilty), we split the data at random into training and test sets: 800 of the records were used for training and the rest (200 records) were used for testing. We examined three variations of the modified LIME Tabular Explainer, corresponding to the three variations presented in section 4: (1) with editing and pruning, which is referred to from here on as "extended LIME", (2) with pruning only, and (3) LIME as-is. For each of these three variations we evaluated their neighborhoods' BP-adherence and explainability adequacy as follows.

5.2.1. Evaluating the Neighborhood's BP-Adherence

We evaluated the quality of the feature perturbations by the adherence of the resulting neighbors to the extracted BP rules in section 4.2. For this comparison, we only considered LIME with editing only (A) and LIME as-is (B), as after the pruning step 100% of the instances adhere to the BP rules. To this end we generated 5000 neighbors per test instance and computed the percentage of neighbors that adhered to the BP rules for each of the two variations. Using LIME as-is, only the test instance to be explained adhered to the BP rules. Due to the way LIME operates, all the generated neighbors did not adhere to BP rule *2.d*. Conversely, the percentage of BP-adhering neighbors using LIME with editing-only greatly varied. These percentages did not follow a normal distribution ($p << 0.01$ in scipy's [16] normality test), so we used a stochastic dominance test [17, 18] with $\alpha = 0.01$ to check for significant difference in the rate of BP-adhering neighbors between the two variations. We repeated the process using four additional random seeds.

5.2.2. Evaluating the Explainability Adequacy

Two of the authors with prior experience in BP modeling have been assigned to independently assess the explanations. For the empirical evaluation, we arbitrarily chose one of the five random

seed's (3) test sets. We extracted only the instances where the top-importance feature was in disagreement between extended LIME and LIME as-is (see Table 2). This set included an extract of 49 application traces. On each application trace, raters have been requested to express what they anticipated a proper explanation would consist of. This included their anticipated outcome (i.e., acceptance or rejection of the loan request) and the feature they considered to have the greatest contribution to that outcome. After evaluating separately, an inner-rater measure of Chronbach's-alpha was computed, reflecting the level of agreement between the two raters.

Since none of the neighbors adhered to the BP rules without editing, the variation of pruning-only was not relevant to this comparison, leaving only two variations: extended LIME and LIME as-is. After dropping all instances that were in dispute between the two raters (two in total), we used the remaining (47) in-consensus instances to compare the raters' anticipated explanations with the two relevant variations.

6. Results

6.1. Business-Process-Adherence of Neighbors

For each of the five random seeds that were used, we ran the stochastic dominance test, comparing the percentage of BP-adhering neighbors with and without editing. The results show that the percentage of BP-adhering LIME-based neighbors with editing-only (A) is indeed stochastically greater than LIME as-is (B) with $p\_value < 0.01$. Moreover, they show it is stochastically greater with a gap of at least 0.40 (40% of the LIME-generated neighborhood) with $p\_value < 0.01$. This means that with BP-aware editing, with high probability, at least 40% of the LIME-based neighbors become BP-adhering.

6.2. Explainability Adequacy

As manual rating is labor-intense, we focused on test instances that had a different top-importance feature in their LIME as-is explanation as compared to their extended LIME explanation. Out of 200 test instances of one of the random seeds (3), for 49 instances the two explanations provided different top-importance features (see Table 2). We then asked the two raters to asses these instances.

The results from the two independent raters for the 49 instances have achieved an overall inter-rater agreement on 47 out of 49 instances (96% agreement). The Chronbach's reliability levels were $\alpha = .937$ for the process results and $\alpha = .939$ for the top-importance feature.

Shown in figure 2 are the explanations for four of the traces that were included in the assessment. Instance (a) in shows an explanation where raters agreed on the top-importance feature being $credit\_score$ (denoted by * and ˆ for raters 1 and 2, respectively). The same feature was also deemed most-important by the extended LIME variation. The LIME as-is variation, however, concluded $Is\_skilled$ was the top-importance feature in its explanations. Instance (b) shows a similar situation to (a), where both raters agreed on $risk$ as the top-importance feature, which was also in agreement with the extended LIME variation, but not not LIME as-is. Instances (c) and (d) reflect the two test instances where the raters did not agree on

Table 2
Cross method agreement regarding top-importance feature.

Top-feature × Top-feature Crosstabulation							
		Extended LIME					Total
		credit_ score	*risk*	*amount*	*Is_ skilled*	*Is_ credit*	
LIME	*credit_score*	67	0	0	0	0	67
	risk	0	84	0	0	0	84
	amount	0	3	0	0	0	3
	Is_skilled	31	15	0	0	0	46
	Is_credit	0	0	0	0	0	0
Total		98	102	0	0	0	200

the top-importance feature (i.e., the two instances that we dropped from the overall set of 49 instances).

We assessed the adequacy of the explanations based on all the 47 instances that were in consensus among the raters. Comparing the top-importance feature according to the raters in these instances, they matched the LIME as-is explanation in 11 instances (23%), while they matched the extended LIME's explanation in 36 instances (77%). For the other random seeds, there were 114, 148, 118, and 128 instances (out of 200 test instances each) for which the two explanation methods (LIME as-is and extended LIME) provided the same top-importance feature.

7. Discussion and Future Research

We presented a BP-model-driven methodology that augments LIME to produce correct and significantly more adequate explanations than the explanations given by LIME as-is.

Contemporary data-driven approaches that infer inter-feature dependencies to provide explanations in the context of BPMSs suffer from the the 'cold-start' limitation (i.e., the inability to give explanations prior to having sufficient process execution historical traces). Our proposed BP-model-driven approach overcomes this limitation.

Our work opens a variety of potential extensions for future research. First, we tested our hypotheses on a relatively small and simple BPs with complete traces. More realistic scenarios in which we also consider "in-flight" instances and the BP rules are automatically generated and tested will make our approach more generic, robust, and scalable. Second, we merely include sequencing constraints as BP-model derivation rules. Inclusion of timestamps and more sophisticated semantics will enrich the scope of our methodology. Third, we do not address global distribution adjustments to the local neighborhood distribution. For example, in our illustrative example (figure 1), we have annotations regarding the global acceptance/rejection rates for the two agent types. These were left out of the scope of our current work but could be part of the local explanations given to specific instances. Forth, to be fully situational-aware, an

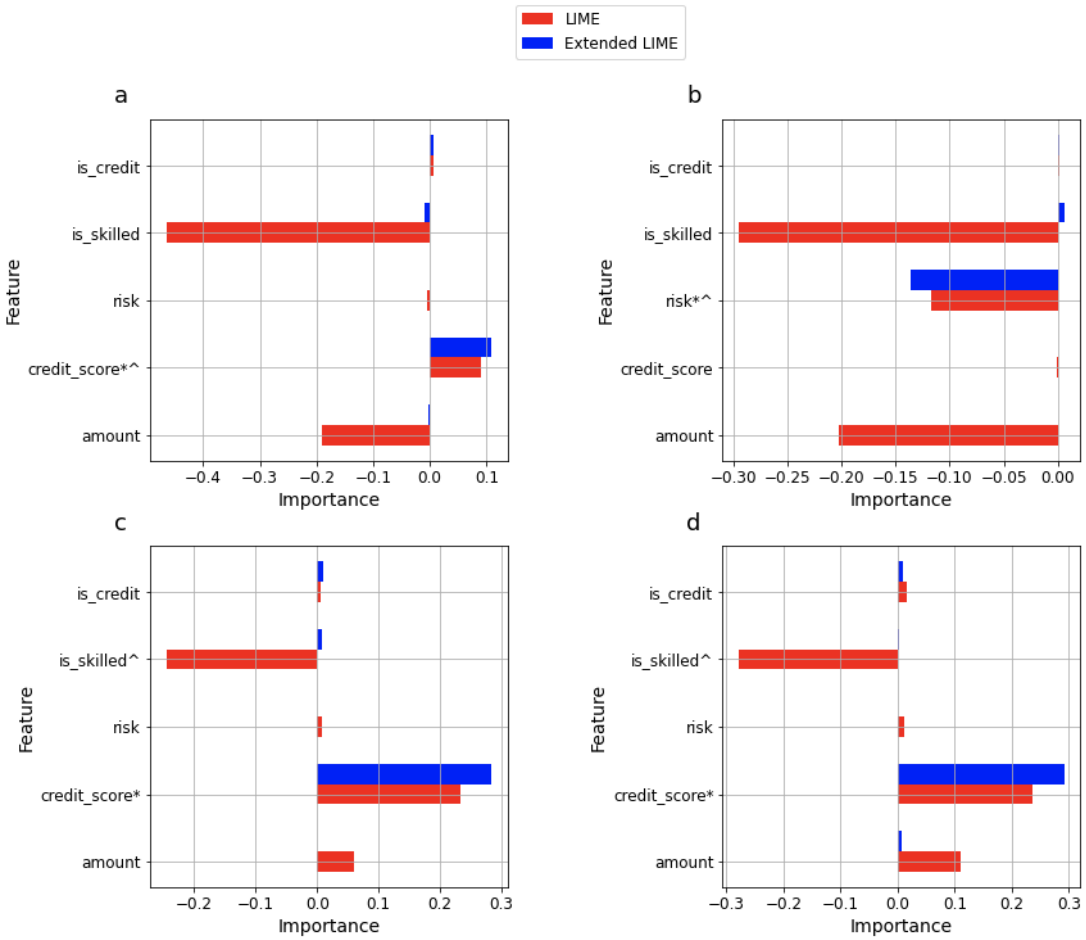

Figure 2: Top-importance features by method for example processes. *a*, *b*, *c*, and *d* correspond to the complete processes described by the four records in Table 1, in order. In red and blue are the importance values assigned to each feature by LIME and extended LIME (with pruning and editing), respectively. The amplitude, regardless of sign, indicates the importance assigned to the feature. * Top-importance features according to rater 1. ^Top-importance features according to rater 2.

ABPMS should also include reasoning about conditional circumstances and implicit rules that are not part of the (explicit) BP model. For example, we suggest such temporal rule annotated for activity "Novice agent" in figure 1, which relates to some temporal implicit changes in the BP execution ("during holiday time").

Acknowledgement

The research was conducted with the support of the State of Israel's Ministry of Aliyah and Integration, Center for Integration in Science.

References

[1] D. Fahland, M. Weidlich, Scenario-based process modeling with greta, in: Proceedings of the Business Process Management 2010 Demo Track, volume 615, CEUR-WS.org, Hoboken, NJ, USA, 2010, pp. 1–6.

[2] A. Deutsch, R. Hull, Y. Li, V. Vianu, Automatic verification of database-centric systems, ACM SIGLOG News 5 (2018) 37–56.

[3] S. T. Jan, V. Ishakian, V. Muthusamy, Ai trust in business processes: The need for process-aware explanations, Proceedings of the AAAI Conference on Artificial Intelligence 34 (2020) 13403–13404.

[4] GDPR, General data protection regulation, 2018. https://gdpr-info.eu/.

[5] CCPA, The california consumer privacy act, 2018. https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB375.

[6] M. Dumas, et al., Augmented business process management systems: A research manifesto, 2022. arXiv:2201.12855.

[7] S. Upadhyay, V. Isahagian, V. Muthusamy, Y. Rizk, Extending lime for business process automation, 2021. arXiv:2108.04371.

[8] M. T. Ribeiro, S. Singh, C. Guestrin, "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, 2016, pp. 1135–1144.

[9] C. Meske, E. Bunde, J. Schneider, M. Gersch, Explainable artificial intelligence: objectives, stakeholders, and future research opportunities, Information Systems Management 39 (2022) 53–63.

[10] A. Adadi, M. Berrada, Peeking inside the black-box: a survey on explainable artificial intelligence (xai), IEEE access 6 (2018) 52138–52160.

[11] S. Verma, A. Lahiri, J. P. Dickerson, S.-I. Lee, Pitfalls of explainable ml: An industry perspective, 2021. arXiv:2106.07758.

[12] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, D. Pedreschi, A survey of methods for explaining black box models, ACM computing surveys (CSUR) 51 (2018) 1–42.

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

[14] S. M. Lundberg, S.-I. Lee, A unified approach to interpreting model predictions, Advances in neural information processing systems 30 (2017).

[15] F. Pedregosa, et al., Scikit-learn: Machine learning in Python, Journal of Machine Learning Research 12 (2011) 2825–2830.

[16] P. Virtanen, et al., SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python, Nature Methods 17 (2020) 261–272.

[17] H. B. Mann, D. R. Whitney, On a test of whether one of two random variables is stochastically larger than the other, The Annals of Mathematical Statistics 18 (1947) 50–60.

[18] R. Dror, S. Shlomov, R. Reichart, Deep dominance - how to properly compare deep neural models, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 2773–2785.