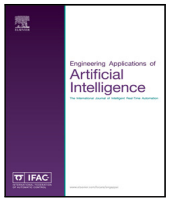
Engineering Applications of Artificial Intelligence 125 (2023) 106678

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai



Generating multi-level explanations for process outcome predictions☆

Bemali Wickramanayake [a,c,*,1], Chun Ouyang [a,c,2], Yue Xu [b,c,3], Catarina Moreira [a,c,d,e,4]

[a] School of Information Systems, Queensland University of Technology, Brisbane, Australia
[b] School of Computer Science, Queensland University of Technology, Brisbane, Australia
[c] Center for Data Science, Queensland University of Technology, Brisbane, Australia
[d] Human Technology Institute, University of Technology Sydney, Australia
[e] INESC-ID/Instituto Superior Técnico, University of Lisboa, Portugal

ARTICLE INFO

Dataset link: https://github.com/bemali/XDD-Net

Keywords:
Business process prediction
Deep neural networks
Event logs
Explainable AI
Machine learning
Process outcome prediction

ABSTRACT

Process mining focuses on the analysis of event log data to build various process analytical capabilities. Predictive process analytics has emerged as one of such key capabilities and it uses machine learning techniques to construct process prediction models. In recent years, deep neural networks have gained increasing interest in process prediction since they can handle multi-dimensional sequential inputs with minimal information loss. However, they are considered black-box models and existing studies in explaining deep neural network-based process predictions rely on only event-level features for explanation. In this paper, we propose a new approach for generating explanations for process outcome predictions at multiple levels. The approach is underpinned by three different prediction models: a transparent model for generating global explanations based on case-level features, an attention-based deep neural network for generating local explanations based on event-level features, and a novel eXplainable Dual-learning Deep network (XD^2-net) for generating local explanations based on case-level features. Using three publicly available datasets, we have tested the applicability of the approach and further examined the multi-level explanations generated by the approach through an elaborate case study. Unlike others, the design of our approach promotes the idea of leveraging the complementary capabilities of different models and utilizing their strengths, rather than focusing on model performance competition. This will contribute towards generating more comprehensive explanations that meet the needs of different end users and purposes in the future.

1. Introduction

Business processes form a lifeline of business within and across organizations. Executions of day-to-day business processes involve a wide range of stakeholders and are supported by a variety of information systems. Data generated along process execution can be extracted from different information systems and curated in the form of *event logs*. An event log consists of event sequences, each of which record the execution of a process instance (a.k.a. *case*) step by step over time. An *event* has a set of *attributes* carrying information of process execution on multiple dimensions, for example, the activity captured by the event, the start or completion time of the activity, the resource(s) who performed the activity, the risk profile of the operation involved in the activity, etc. Hence, event logs are considered multi-dimensional time sequence data and often capture rich context information about process execution.

Process mining focuses on the analysis of event log data to build various process analytical capabilities (van der Aalst, 2016). In recent years, predictive process analytics has been developed as one of such capabilities, where machine learning techniques are being applied to construct *process prediction models*. These models aim at predicting future states of an ongoing case by learning from the process execution history recorded in event log data. A typical example of process prediction is predicting outcomes that the execution of a business process may lead to, known as *process outcome prediction*. It can be used to predict stage-wise outcomes for an end-to-end process that consists of several stages with stage-specific outcomes (Le et al., 2014), and

☆ This document presents findings from a research project funded by Queensland University of Technology, Australia.
* Corresponding author at: School of Information Systems, Queensland University of Technology, Brisbane, Australia.
E-mail addresses: bemali.wickramanayake@hdr.qut.edu.au (B. Wickramanayake), c.ouyang@qut.edu.au (C. Ouyang), yue.xu@qut.edu.au (Y. Xu), Catarina.pintomoreira@uts.edu.au (C. Moreira).
[1] 0000-0001-6480-3513
[2] 0000-0001-7098-5480
[3] 0000-0002-1137-0272
[4] 0000-0002-8826-5163

https://doi.org/10.1016/j.engappai.2023.106678
Received 30 December 2022; Received in revised form 5 June 2023; Accepted 16 June 2023
Available online 8 July 2023

also to predict decision outcomes for a process involving those decision points of which the output decisions can be clearly classified (Hsieh et al., 2021). Hence, process outcome prediction is considered widely applicable in practice and a benchmark of the performance of various process outcome prediction models can be found in Teinemaa et al. (2019).

In a machine learning-based approach, event logs are converted to trainable features and used as the input data to train prediction models. Whilst advanced machine learning models such as gradient-boosted trees can achieve good performances in process outcome prediction (Tama and Comuzzi, 2022), deep neural networks such as long short-term memory (LSTM) networks, have gained increasing interest in process prediction (Evermann et al., 2017; Camargo et al., 2019; Metzger et al., 2019) since they can inherently handle multi-dimensional sequential inputs (e.g., as tensors) so to preserve the execution order of process activities as well as to capture the relevant context data thus ensuring minimum information loss during feature construction for better prediction accuracy.

However, models that have complex internal architectures, such as deep neural networks, are considered black-box models because their reasoning for learning decisions is not transparent. Explainable AI (XAI) aims to address the issue of black-box models, and two mainstream approaches have been proposed. One is called post-hoc explanation, which leaves the complicated model as it is and attempts to explain the model predictions by using simpler surrogate models. Whilst surrogate models can be used to approximate how the original model arrived at its decision, they are limited to estimating how a black-box model made its decision based on the inputs and outputs. As a result, explanation fidelity (faithfulness to the original model) becomes one of the key criteria for evaluating post-hoc explanations. In another approach, known as an intrinsic explanation, a transparent model (e.g., linear or logistic regression) or partially black-box model that is incorporated with certain explainable mechanisms (e.g., attention-based LSTM) and hereafter referred to as an *intrinsically explainable model*, is used to make a prediction and generate explanation about the model decision. It has been suggested that a transparent or intrinsically explainable model is more appropriate to use than a surrogate model, especially for explaining high-stake decisions (Rudin, 2019).

So far, a number of studies in process mining have employed model attention mechanisms in deep learning-based process prediction models for generating intrinsic explanations (e.g., Sindhgatta et al., 2020a and Wickramanayake et al., 2022a). Although these models are capable of handling event sequences with minimal information loss, their output has to rely on *event-level* features for explanation. Such explanation often presents information of low granularity, e.g., 'occurrence of activity A at timestep 3 and/or timestep 4 has a certain influence on the model's decision in rejecting a loan application'. This may be useful for a data scientist (to conduct model inspection) but can be difficult for a business user to extract insights. A business user, such as a process analyst, is usually interested in *case-level* features captured by information consolidated from event-level attributes. For example, an explanation stating that 'the number of activity A's occurrences in processing a loan application has the highest impact on the model's decision in rejecting the application' would be more intuitive and useful to inform strategies for reducing loan rejection rate. Among others, how to generate explanations using case-level features from a process prediction model built upon deep neural networks has yet to be addressed.

In this paper, we are interested in generating explanations for process outcome predictions at multiple levels. The meaning of 'multiple levels' is twofold. One refers to the construction and representation of features at the *event-level* and *case level* from the aspect of process execution, whereas the other refers to explanations at the *local* and *global* levels from the aspect of XAI. In XAI, an explanation is essentially an answer to why and how a model arrived at its decision (Miller, 2019), and different levels of granularity in explanations offer this answer differently (Wickramanayake et al., 2022b). *Global explanations* are those that would offer a common explanation for the prediction of any data point (e.g., an entire collection of process instances), and a *local explanation* is specific to the prediction of a given data point (e.g., a specific process instance). Hence, global explanations are useful to gain an overall understanding of a model's decision making, whereas a local explanation can be used to extract insights into how the model arrived at its decision in a specific instance.

To this end, we propose a machine learning-based approach for generating multi-level explanations for process outcome predictions. Our approach is underpinned by three different prediction models: a transparent model for generating global explanations based on case-level features, an attention-based deep neural network for generating local explanations based on event-level features, and a new model named *eXplainable Dual-learning Deep network* (XD^2-net) for generating local explanations based on case-level features. In XD^2-net, the novelty is that the model is trained using event-level attributes to learn the weights of case-level features. The design of such internal architecture allows the model to generate explanations using case-level features based on the input of event-level attributes that minimizes information loss. Also, it is worth noting that our approach promotes the idea of leveraging the complementary capabilities of different models and utilizing their strengths, rather than focusing on model performance competition. This will contribute towards generating more comprehensive explanations that meet the needs of different end users and purposes in future research.

The rest of the paper is organized as follows. Section 2 provides an overview of explainable AI, existing techniques of explaining a model, current work in explaining process predictive models and evaluation of model explanations. Section 3 details our approach for generating multi-level explanations for process outcome predictions and the proposed ensemble architecture of XD^2-net. Section 4 presents the experimental setup and results of the quantitative evaluation of the explanations and explainable models. Section 5 examines the multi-level explanations that are generated through our approach, along with a quantitative evaluation of interpretability of global explanations. Finally, Section 6 summarizes the contributions of our work and outlines future work.

2. Related work

In this section, we discuss the notion of model explainability, the existing techniques for explaining models, and the evaluation of model explanations. Further, we introduce how model explainability is applied in the application domain of predictive process analytics.

2.1. Explainable AI

The primary need for explaining a machine learning model is to understand what are the criteria/approach used by the model to arrive at a decision. However, this basic requirement is expected to fulfil a range of end objectives (Nunes and Jannach, 2017), which can be listed as follows; Transparency, Effectiveness, Trust, Persuasiveness, Satisfaction, Education, Efficiency, Scrutability, and Debugging.

A model can be classified as either a *transparent model* or a *black-box model* in terms of their explainability (Guidotti et al., 2019). A model is called transparent when its decision-making process is entirely visible via feature weights or heuristics. Regression models, Bayesian networks, and Decision Trees are examples of such transparent models. A black-box model (which is, in contrast, not transparent) can be explained by either approximating the model's performance/outcomes by an explainable surrogate model/mathematical relationships or by making the model transparent to explain the model using its internal properties. The first method is called *post-hoc explanation*, and the second is called *intrinsic explanation*. The most popular post-hoc methods include LIME (which tries to establish a linear relationship between

input and output locally) (Ribeiro et al., 2016), SHAP (which determines a feature's contribution towards the model decision using shapely values) (Lundberg and Lee, 2017), and the use of transparent surrogate models to approximate cohorts of the input space. A model can be intrinsically explained either by using properties of the underlying model to fully or partially explain the model, such as using regression activation maps (Wolanin et al., 2020) (to explain convolutional neural networks), and attention weights (Choi et al., 2016) to explain long short-term memory (LSTM) networks. Another approach to explaining computationally complex models intrinsically is to incorporate explainability to the model in the design of the model itself. This is done by augmenting the complex model with an explainable white box (Alvarez-Melis and Jaakkola, 2018; Chen et al., 2018; Kraus and Feuerriegel, 2019), and explaining the model completely via the white box. Hence, such models can be called semi-white boxes.

Model explanations are and need to be oriented towards the intended users of those explanations (Ribera Turró and Lapedriza, 2019; Tomsett et al., 2018). The particular purpose and the end user of the explanation would guide how the explanations need to be generated and presented (Chromik and Schuessler, 2020; Wickramanayake et al., 2022b). Thus, when the model explanation is as critical as the model performance for a particular application, instead of involving the user at the final stage getting involved in the design of the explanation (hence often the model) may lead to better explainability (Hoque and Mueller, 2022; Kwon et al., 2019).

2.2. Explaining deep learning-based process predictions

Predictive process analytics is a relatively new branch of data analytics dedicated to providing business process intelligence in modern organizations. It uses event logs, which capture process execution traces in the form of multi-dimensional sequence data, as the key input to train predictive models. These predictive models, underpinned by advanced machine learning or deep learning techniques, can be used to make predictions about states of business process execution. In particular, recurrent neural networks (RNNs) and their variants such as long short-term memory (LSTM) networks have naturally found applicability in predictive process analytics. For example, given an input event log that records the sequence data of a running business process, an LSTM-based model can be trained to predict the next event in the running process (Evermann et al., 2017; Tax et al., 2017; Camargo et al., 2019) as well as the remaining execution time of the running process (Camargo et al., 2019).

If exploratory process model-based techniques or transparent machine-learning techniques (e.g., Logistic regression/decision tree) are used for the process predictions, how the model makes prediction is transparent to the user. However, when deep learning-based techniques are used, how those predictions were made is opaque to the user. Thus, these models require to be explained to gain that visibility. Most of the current work in *explainable* deep learning-based predictive process analytics approaches use post-hoc methods such as LIME and SHAP to explain the model's prediction (Sindhgatta et al., 2020b; Galanti et al., 2020; Mehdiyev and Fettke, 2020), while some of the recent approaches focus on intrinsic interpretable deep learning architectures. These include the approaches that use model attention (Sindhgatta et al., 2020a; Wickramanayake et al., 2022a), and use of partial dependence plots (Mehdiyev and Fettke, 2021), and Layer wise relevance propagation (Samek et al., 2019; Weinzierl et al., 2020). A semi-white box proposed in this space is the approach of combining the process model with a gated graph neural network (Harl et al., 2020).

Post-hoc methods such as LIME and SHAP are model agnostic, hence can be used irrespective of the underlying model architecture to explain a deep-learning-based process prediction model. However, the fidelity (faithfulness) of these explanations towards the underlying predictive model is a concern and needs to be evaluated before using them (Velmurugan et al., 2020, 2021). On the other hand, intrinsic approaches do not encounter this problem but given the complexity of the deep-learning architecture, these explanations may not completely explain the model. Semi-white boxes are also an intrinsic explainable approach, however, the difference is that the explainability is embedded in the architecture itself to a greater extent, by combining a white-box with a deep-learning-based black-box. Although in these techniques, the prediction could be explained completely using the white-box component, it still does not explain the black-box's decision mechanism.

2.3. Evaluating model explanations

A model explanation can be evaluated using two main criteria (Guidotti et al., 2019). *Interpretability* refers to how easy an explanation is to be understood by a human and *fidelity* refers to how truthful the explanation is to the model being explained. Since the post-hoc explanations are external to the actual model of interest, there arises the need to evaluate how truthful is the explanation towards the model (Zhou et al., 2021) and requires specific evaluation measures and metrics of 'explanation fidelity' (Velmurugan et al., 2020). If a model is transparent or intrinsically explainable such a need does not arise, as such explanations are inherently truthful to the model and are evident to be as such with experiments (Samek et al., 2019). However, intrinsic explanations still need to be evaluated for explanation interpretability (Stevens et al., 2022). Explanation interpretability can be evaluated either using application- or human-grounded techniques or functionally grounded techniques (Doshi-Velez and Kim, 2017).

Application- or human-grounded evaluation refers to evaluating the explanation based on human-centric experiments. The primary difference between application and human-grounded techniques is that for application-grounded evaluations experiments are conducted with an 'expert' team and human-grounded evaluations are conducted using the opinions of laypeople. In these experiments, we can use qualitative metrics (e.g.: usefulness, satisfaction, trust) as well as quantitative metrics (e.g.: time consumed to grasp the explanation, recall of explanation) (Narayanan et al., 2018). Functionally grounded metrics for evaluating model interpretability are derived by further formulating the attributes of an explanation that are evident to be important for humans to grasp an explanation effectively. Such metrics include effective complexity/conciseness, explanation (dis)agreement and explanation continuity.

Effective complexity/conciseness (Nguyen and Martínez, 2020) evaluates the minimum number of features that are 'required' to explain a model decision, where the intuition behind this metric is that the lower the number of features that are used to explain, the easier for the user to grasp and recall the explanation. There is a series of metrics proposed to measure explanation agreement (Krishna et al., 2022) that are used to evaluate the level of agreement between two or more explanations, in terms of the top 'k' number of features each explanation considers as important. The intuition behind this metric is that users of explanations are mostly concerned about top 'k' features, often $k = 1$ or $k = 2$. Explanation continuity (Montavon et al., 2018) is defined as the prevalence of a continuous explanation function, and in simpler terms, the requirement for similar inputs to have similar explanations, which helps the users of the explanation to trust the explanations as they do not drastically vary for similar inputs.

3. Approach

Fig. 1 depicts our approach for generating multi-level explanations for process outcome prediction. The approach takes an event log as the input, which is first truncated into process prefixes at the predefined prediction points. Then, two sets of features are extracted from these process prefixes: *case-level* features and *event-level* features. Three independent model architectures are then trained with these features where each of which make the outcome prediction along with a different level of explanation.

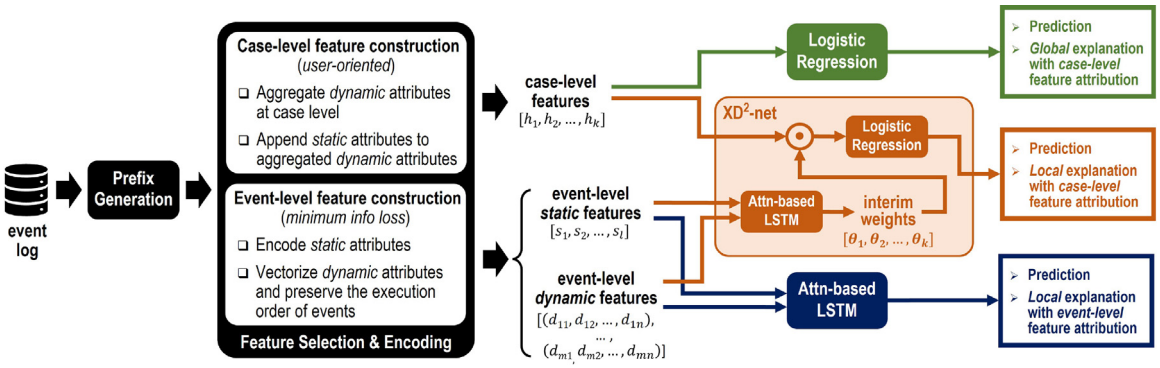


Fig. 1. Proposed approach: Three independent model architectures that use two levels of feature vectors to predict the process outcome and generate three levels of intrinsic model explanations.

1. Logistic regression model: trained with case-level features and generating a global explanation with case-level feature attribution, which is an explanation generalizable for all the predictions, generated using the features that are representative of a given case. E.g.: Contribution of the number of times the activity A was executed over the process trace towards the decision of the model = 0.5.
2. Attention-based LSTM model: trained with event-level features and generating local explanations with event-level feature attribution, which is an explanation specific to a given prediction of a given case, using the features that are specific to an individual event. E.g.: Contribution of the execution of activity A at event 2 towards the prediction for case ID C = 0.2.
3. An ensemble architecture of the above two models which has led to the design of eXplainable Dual-learning Deep network (XD^2-net): trained with event-level features, learning weights of case-level features, and generating local explanation with case-level feature attribution, which is an explanation specific to a given prediction, using the features that are representative of the entire process prefix and the case attributes of a given case. E.g.: Contribution of executing activity_A 3 times over the process trace towards the prediction for case ID C = 0.4.

A detailed description of each component of this approach is as follows (see Fig. 2).

3.1. Terminologies and notions

An *event log* refers to the recorded form of process execution, which has an entry per each event of each process execution instance. The entries of an event log are arranged based on the concept of a *process trace*. A process trace captures the execution of a single process instance, identified by a case identifier *Case ID* and ordered by the execution order of process *events*. Each event recorded in the event log has a Case ID as the process instance identifier and event-specific attributes. The mandatory event attributes that must be present in any event log are *activity* that was executed at the given event, and *timestamp* of the execution. Most event logs contain additional event-specific attributes that include the *resource* who executed the activity, and an *event transition label* which indicates if a given activity is in triage, in progress, or completed. Some event logs also record certain case-specific attributes as well, that remain static throughout the process trace. A process *prefix* is a partial process trace, that has the same beginning as the original trace but is truncated before the end of the trace. Formal mathematical definitions of the above are as follows, which will help us to formalize the construction of XD^2-net.

Definition 1 (*Trace van der Aalst, 2016*). A *trace* is a non-empty sequence of unique events e_i with a common case identifier c. Let $m = |\tau|$ and $\tau = [e_1, \dots, e_m]$ is defined as a process trace, where l is the length of the process trace. For all $i, j \in \{1, \dots, l\} : c_{e_i} = c_{e_j}$ For $1 \le i < j \le n$: $e_i \neq e_j$ (i.e., each event appears only once), and $t_{e_i} \le t_{e_j}$ (i.e., the ordering of events in a trace should respect their timestamps).[5]

Definition 2 (*Prefix van der Aalst, 2016*). A *prefix* is a partial process trace with the same beginning as the original trace. Let $m' = |\rho|$ and $\rho = [e_1, \dots, e'_m]$ is defined as a process prefix, where m' is the length of the process prefix and $m' < m$.

Definition 3 (*Event-Specific Attributes*). Let C be the set of case identifiers, and D^i a set of values that belongs to a particular (dynamic) event-specific attribute, where $i \in [1, 2, \dots, n]$ if the event log records n number of event-specific attributes. $\mathcal{E}$ is the set of *events*, and each event has the above *attributes*. For any $e \in \mathcal{E}$: $c \in C$ is the case identifier of event e, $d^i_e \in D^i$ which is an attribute specific to the event e, which belongs to the set of such attributes D^i.

Remark. Examples of event-specific attributes are the activity performed at a given event, the resource who performed the activity for a given event, and the time at which the activity was completed for a given event.

Definition 4 (*Case Specific Attributes*). Let C be the set of case identifiers, and S^i a set of values that belongs to a particular (static) case-specific attribute, where $i \in [1, 2, \dots, l]$ if the event log records l number of case-specific attributes. Each case in C has the above *attributes*. For a given $c \in C$, $s^i_c \in S^i$ which is the identifier that belongs to the case-specific attribute S^i.

Remark. An example of case-specific attributes is loan application category. Note that the case-specific attributes however should not be confused with the case-level features (see below).

Finally, we introduce two more important concepts.

- *Event-level features* refer to the features that are derived from the unique attributes associated with a specific event within a process prefix. These features are constructed by capturing event-specific attributes that provide a detailed description of the event itself.
- *Case-level features* are constructed by combining event-specific and case-specific attributes to provide a comprehensive description of the process prefix and the associated case.

3.2. Case-level and event-level feature construction

We begin our approach by generating prefixes through truncation of process traces at predetermined prediction points, which are specific events in the process trace where predictions about the process

[5] Event index numbers take precedence over timestamps where two events occur concurrently.

outcome are made. Further details on our experimental approach for determining the prediction points and prefix generation can be found in Section 5. Once we have the prefixes, we move on to the feature encoding phase. In this phase, from the generated prefixes we extract two sets of features. The first set of features, called event-level features, describes the characteristics of a particular event within a given prefix. The second set of features, called case-level features, provides a holistic description of the entire prefix and the associated case.

Event-level feature set consists of the process prefixes' event-specific attributes (e.g.: Activity label, Resource label) which are encoded into a sequential feature vector and arranged as a 2-D feature vector per case identifier, where the first dimension is the event and the second dimension is the feature value. We can mathematically represent the 2-D event-level feature vector $\boldsymbol{F_d}$ as follows.

$$\boldsymbol{F_d} = \begin{bmatrix} d_{11}, & d_{12}, & ...., & d_{1n} \\ d_{21}, & d_{22}, & ...., & d_{2n} \\ ..... & & & \\ d_{m1}, & d_{m2}, & ...., & d_{mn} \end{bmatrix}$$

where n is the number of distinct event-specific features, and m is the prefix length.

In contrast, the case-level feature vector (denoted by vector $\boldsymbol{H}$) consists of an aggregated form of event-level feature vector $\boldsymbol{F_{d-agg}}$ and a case-specific feature vector $\boldsymbol{F_s}$.

The 2-D event-level feature vector is aggregated over the event dimension into an aggregated 1-D feature vector as follows. The aggregation will be performed using an appropriate aggregation technique for each feature, which can be sums, counts, averages, medians etc.

$$\boldsymbol{F_{d-agg}} = AGG_m(\boldsymbol{F_d})$$

$$\boldsymbol{F_{d-agg}} = \begin{bmatrix} d-agg_1, & d-agg_2, & ..., & d-agg_n \end{bmatrix}$$

The case-specific feature vector consists of the case-specific attributes (e.g.: Loan application type, Age of the applicant), which are encoded into a 1-D feature vector ($\boldsymbol{F_s}$) per case as follows.

$\boldsymbol{F_s} = \begin{bmatrix} s_1, & s_2, & ..., & s_l \end{bmatrix}$ where l is the number of case-specific attributes.

The case-level feature is then constructed by adjoining the aggregated event-level feature vector $\boldsymbol{F_{d-agg}}$, with the case-specific feature vector $\boldsymbol{F_s}$, we denote the case-level feature vector $\boldsymbol{H}$ as follows.

$$\boldsymbol{H} = \boldsymbol{F_{d-agg}} \oplus \boldsymbol{F_s}$$

$\boldsymbol{H} = \begin{bmatrix} h_1, & h_2, & ..., & h_k \end{bmatrix}$ where k is the total number of case-level features.

3.3. Model architecture

To derive each level of explanation that explains the process outcome prediction, we deploy three models, as depicted in Fig. 1. On the top is a logistic regression model, which takes the case-level feature vector as the input and makes the prediction along with global explanations using case-level feature attribution. At the bottom is an attention-based LSTM architecture, which takes the event-level feature set (which ensures minimal information loss) as the input and makes the prediction along with local explanations using event-level feature attention weights (Sindhgatta et al., 2020a; Wickramanayake et al., 2022a). In the middle is an ensemble model that takes both event-level and case-level feature sets as the input to predict the outcome of the process and generates local explanations using the case-level feature set. The model employs a dual-learning mechanism as inspired by the work in Alvarez-Melis and Jaakkola (2018) and takes the simpler

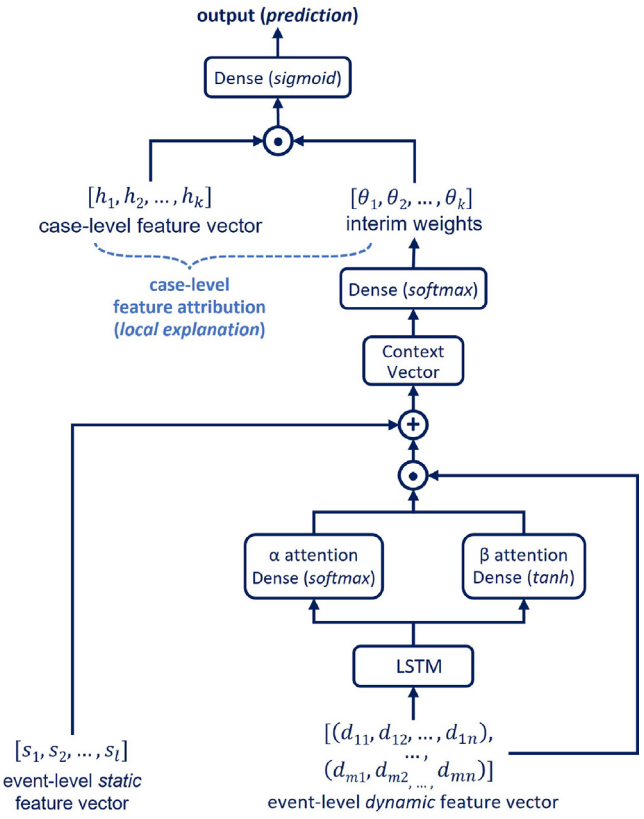


Fig. 2. Model architecture of XD^2-net.

case-level feature set for an explanation. It uses the attention-based LSTM architecture as a computational back end, the transparent logistic regression model for generating explanations. Moving forward, we refer to this model as *eXplainable Dual-learning Deep Network (XD^2-net)*. Further, despite XD^2-net explaining the model prediction with the lossy case-level feature set, it is expected to compensate for the loss of information via the event-level feature set, which is fed to its LSTM back end.

XD^2-net consists of two components. An LSTM-based computational engine that accepts an event-level feature vector and outputs a set of weights that are equivalent to the number of case-level features (k) and a decision layer that multiplies those weights with case-level features and gives the final output via a dense layer with the sigmoid activation function (where the final layer is equivalent to a logistic regression model).

Computation of feature weights: The LSTM-based computational engine consists of two parts, a bi-directional LSTM combined with two attention mechanisms (α and β) and a final dense layer. The attention mechanism is introduced to improve the model performance, however, one can use the output of these attention layers (attention weights) to examine the contribution of event-level features towards the model output as well (Sindhgatta et al., 2020a).

The bi-directional LSTM accepts the dynamic event-level feature vector $\boldsymbol{F_d}$. This feature vector carries the event-level information, preserving the order of the event sequence. Bi-directional LSTM computes the relationship between these events, both in forward and backward directions. For each sequence that is passed through the bi-LSTM, α attention mechanism computes the importance of each event (e) and β attention mechanism computes the importance of each attribute (d) in each event (e). Then, the dynamic event-level feature vector $\boldsymbol{F_d}$ is element-wise multiplied by the two attention vectors and summed across the sequence length (prefix length) m to generate the intermediate vector $\boldsymbol{F_d^{int}}$. This intermediate vector then carries the information

about the original feature vector, the relative importance of each feature value in a given event, and the relative importance of each event.

$$\boldsymbol{F}_{\boldsymbol{d}}^{\boldsymbol{int}} = \sum^{m} \alpha \odot \beta \odot \boldsymbol{F}_{\boldsymbol{d}}$$

Then, we append the static feature vector $\boldsymbol{F}_{\boldsymbol{s}}$, to the computed $\boldsymbol{F}_{\boldsymbol{d}}^{\boldsymbol{int}}$ and send this through a two-layer dense network with the activation function softmax and the weight matrix $\boldsymbol{W}$, to generate the interim weights $\boldsymbol{\theta}$.

$$\vec{\theta} = softmax(\boldsymbol{W}^{T}(\boldsymbol{F}_{\boldsymbol{d}}^{\boldsymbol{int}} \cup \boldsymbol{F}_{\boldsymbol{s}}))$$

where $\vec{\theta} = [\theta_1, \theta_2, \ldots, \theta_k]$

The non-linear relationships that are introduced by the bi-LSTM and the dense network are expected to identify the distinguished patterns that exist in the input process prefixes that may not be captured at the case-level feature vector level.

Final prediction: The decision layer consists of the following components. (1) A multiplication layer that multiplies the computed interim weight vector $\boldsymbol{\theta}$ with the case-level feature vector $\boldsymbol{H}$, (2) A dense layer that accepts the multiplied vector and uses a *sigmoid* function to predict the probability of the outcome.

The complete model can be mathematically expressed as follows.

$C = \boldsymbol{\theta} \odot \boldsymbol{H}$; dot multiplication of the case-level feature vector and the vector of their interim weights.

$$P(y) = \frac{1}{1 + e^{-(b + w \odot C)}} \text{ ; final output}$$

During the training phase, the model trains by comparing the final output against the actual outcome for each training sample using binary cross-entropy. Via training, the LSTM-based engine learns to identify the patterns in the dataset and provide the best set of weights that aids the final model decision.

Extraction of explanations: The case-level feature vector that enters the final sigmoid activation function is multiplied by two weight vectors; $\boldsymbol{\theta}$ - the interim weight vector, and w - the final dense layer weight vector. Thus, the feature importance for each feature h_i is extracted by multiplying the ith weight of $\boldsymbol{\theta}$ and w.

Feature importance of $h_i = \theta_i \odot w_i$

Out of these two sets of weights, whilst w is static (once trained), $\boldsymbol{\theta}$ is a computed vector of weights by the LSTM-based backend unique to each data instance provided to the model. Therefore, the final feature importance computed ($\theta_i \odot w_i^1$) is also instance specific (i.e., local).

4. Experiments and evaluation

This section outlines our experimental setup and presents the findings from our empirical evaluation of the three prediction models utilized in our proposed approach for explainable process outcome prediction. The experimental source code can be obtained from the following link: https://github.com/bemali/XD2-Net.

During the experiments, we assessed the effectiveness of the logistic regression, attention-based LSTM architecture, and XD^2-net models in terms of both the quality of predictions and explanations. These models were employed in the approach described in Section 3.

4.1. Datasets and preprocessing

We evaluate model performance using three publicly available event log datasets. These datasets capture processes in different application domains. BPIC 2012 (van Dongen, 2012) and BPIC 2017 (van Dongen, 2017) both represent a loan application process from a Dutch financial institution, where BPIC 2017 represents a more streamlined version of the BPIC 2012 process.BPIC 2018 represents an agricultural grant application process, with data for three consecutive years from 2016 to 2018. BPIC 2018 log is organized into eight sub-logs each representing a different documentation category. When all eight logs were combined, the process represented in BPIC 2018 displays concept drift, as a result of the geo-parcel document log replacing certain key activities in the year 2017 that were previously fulfilled by the activities in parcel document and entitlement application logs in previous years (Denisov et al., 2018). Concept drift pertains to the phenomenon wherein the underlying data distribution undergoes alterations over time, rendering a previously trained model inadequate for generating precise predictions (Demšar and Bosnić, 2018). Within the context of BPIC 2018 (full log), concept drift assumes a notable degree of significance due to the dual occurrence of modifications in the distribution of activity labels and the complete replacement of one set of activities with another throughout the temporal progression.Thus, for our experiment, we consider only the payment application log of BPIC 2018, which does not exhibit concept drift.

Our first step in transforming these event logs appropriate for predicting outcomes was to determine a relevant outcome for each process. Next, we truncated the process traces to only include the events that happened prior to the identified outcome. Additionally, we performed distinct pre-processing procedures for each log. Below outlines the measures we took to identify the outcomes and conduct the particular pre-processing tasks.

BPIC 2012: The loan application process has five outcomes, namely A_CANCELLED, A_DECLINED, A_APPROVED, A_ACTIVATED, and A_REGISTERED. A_CANCELLED and A_DECLINED represent an unsuccessful outcome, while A_APPROVED, A_ACTIVATED, and A_REGISTERED represent a successful outcome. In order to simplify the analysis, we combined A_CANCELLED and A_DECLINED into a single outcome called A_UNSUCCESSFUL, and A_APPROVED, A_ACTIVATED, and A_REGISTERED into a single outcome called A_SUCCESSFUL. The activity label for the log was determined based on the field 'CONCEPT_NAME', and only the events with the life cycle transition marked as 'complete' were considered. We also merged O_CANCELLED and O_DECLINED activities and removed three redundant activities from the log (Bautista et al., 2013). Finally, for explainability purposes, we translated the activity labels that were originally in Dutch into English.

BPIC 2017: This log represents a more improved version of the same process as BPIC 2012 log. In this process, there are three outcomes, A_DENIED, O_REFUSED and O_ACCEPTED. O_REFUSED outcome is a redundant outcome which is always followed by A_DENIED, hence we combined the two outcomes together as A_DENIED.

BPIC 2018 (application): This log has three sub-processes application, objection and edit. Out of these sub-processes, the application subprocess is the primary process whereas the other two sub-processes are outliers (Denisov et al., 2018). To arrive at the activity label, we combine the fields 'subprocess' and 'activity2'. Except for 6 cases, all the cases in this log end up in the outcome 'application_finish payment'. However, most of the traces encounter the event 'application_abort payment' after the payment was initiated (activity 'application_begin payment'). Thus, we chose if the application goes through the activity 'application_abort payment' or directly goes into the outcome 'application_finish payment' as our outcome.

Table 1 provides some details of these preprocessed event logs.

4.2. Prediction points and prefix generation

Each process trace that is present in the event log first needs to be truncated to generate the prefixes that are used to train the model. The point at which the process trace is truncated to generate the prefix then also becomes the point at which the model makes the prediction. To

Table 1
Details of the event logs used in experiments.

Event Log	Process	# cases	Avg. case length	Max. case length	Dynamic attributes	Static attributes	Predicted outcome	Detailed outcome
BPIC 2012 (van Dongen, 2012)	Loan application	12 688	9	85	Activity Resource Timestamp	Loan amount	A_SUCCESSFUL or A_UNSUCCESSFUL	If the loan application is approved or declined
BPIC 2017 (van Dongen, 2017)	Loan application	20 980	14	47	Activity Resource Timestamp Credit score	Loan amount requested Number of Term Application type	O_ACCEPTED or A_DENIED	If the loan offer was accepted or application was denied
BPIC 2018 (Application) (van Dongen and Borchert, 2018) payment appl.	Agriculture grant application	43 704	16	52	Activity Resource Timestamp	Amount applied	application_finish payment or application_abort payment	If grant was paid on time or not

determine an appropriate prediction point, we consider the following criteria which are based on the traditional data mining criteria of dataset size, feature representation and dataset balance that are used in machine learning-based predictive analytics.

- **Criterion 1 - a sufficient number of data samples:** When the prediction is made, there is a sufficient number of traces to train the model effectively
- **Criterion 2 - sufficient coverage of data variants:** When the prediction is made, the process is required to be progressed well enough that there is sufficient information (diversity among traces) to make predictions
- **Criterion 3 - data balance:** the traces are balanced between the outcome targets. This criterion may not always be satisfied, as there are certain processes that naturally exhibit an imbalance in outcomes, regardless of the decision point. Traditionally, to balance a dataset, under-sampling or oversampling techniques are used, but each of these techniques has its own drawbacks. Heavy under-sampling can introduce bias to the dataset and oversampling techniques like SMOTE may not be suitable to impute new samples for event logs that represent sequences of events, which can end up generating samples that cannot exist in the actual process.

Exploratory data analysis is employed to determine if the criteria are met by analysing the number of process traces by the end outcome (to assess criteria 1 and 3) and the number of process variants (to assess criterion 2) along the process trace. After establishing the prediction point (event m) to generate a prefix, we sort all events in the process trace by activity index or timestamp. We then truncate the process trace at event m, where $m < l$ (trace length), by considering only the first m events.

The number of cases belonging to each outcome for every log, categorized by prefix length, is shown in Fig. 3. However, for BPIC 2012, prefixes shorter than 5 events show an unbalanced distribution of outcomes (criterion 3), and prefixes shorter than 20 events have insufficient cases for model training (criterion 1). Additionally, prefixes shorter than 5 events make up a small fraction of the total variants in the event log as depicted in Fig. 4 (criterian 2). Therefore, prediction points 5 to 20 are chosen for generating prefixes in BPIC 2012.

Similarly, for BPIC 2017, prediction points 12 to 20 are used, and for BPIC 2018, prediction points 12 to 28 are utilized. This method ensures that there is sufficient data for training the model and that the selected prefixes are representative of the overall event log.

4.3. Preprocessing of prefixes

At each prediction point, the generated prefix set is split into two parts: training (70% of cases) and validation (30% of cases). In cases where the datasets are unbalanced with regards to the distribution of the prediction target (outcome), the majority target was undersampled to balance the training set. Since prediction points with minimal outcome imbalance are selected, undersampling will have a minimal impact on the training dataset size. The test set was not balanced to evaluate the model performance.

For larger organizations, the 'resource' attribute may contain many labels representing the 'name' or 'ID' of each individual involved in the process. However, many of these resources can be combined into a few roles based on the tasks they perform. To accomplish this, a role discovery algorithm is used to cluster resources based on the frequency of a particular activity they perform, resulting in a smaller number of roles from the numerous resource labels in the event log (Zhao and Zhao, 2014).

4.4. Feature selection and encoding

For the experiments, we construct two feature sets for each event log. An event-level feature set (for XD^2-net and LSTM models) and a case-level feature set (for XD^2-net and logistic regression models).

4.4.1. Event-level feature vectors

The event-level feature vector consists of the event-specific attributes of the event log, which change over the execution of a given process trace of a given case.

To construct the event-level feature vector, the event-specific attributes of the activity label, resource label (now converted to a role label), timestamp and any relevant secondary event-specific attribute available in the log (e.g.: event-level credit score in BPIC 2017) were considered. With the timestamp attribute, two temporal attributes are generated; 'time elapsed' - the total time elapsed from the beginning of the trace to a particular event and 'task duration' - the time gap between the completion of a given event and the completion of the previous event. Further, All categorical attributes were one-hot-encoded to convert into numerical features, and all numerical features were min–max normalized. For a given trace, the event-specific feature vector is arranged as a sequence of features, arranged by the order of the events, hence representing a two-dimensional feature vector for a given case identifier.

4.4.2. Case-level feature vector

Construction of the case-level feature vector is started with the already constructed event-level feature vector. First, the event-level feature vector is aggregated along the event dimension, resulting in a flat feature vector. For the aggregation of the features, we use count of occurrence as the aggregation method for activity labels and role labels representing the number of times a particular activity has occurred or a particular role was involved in the process over the entire process

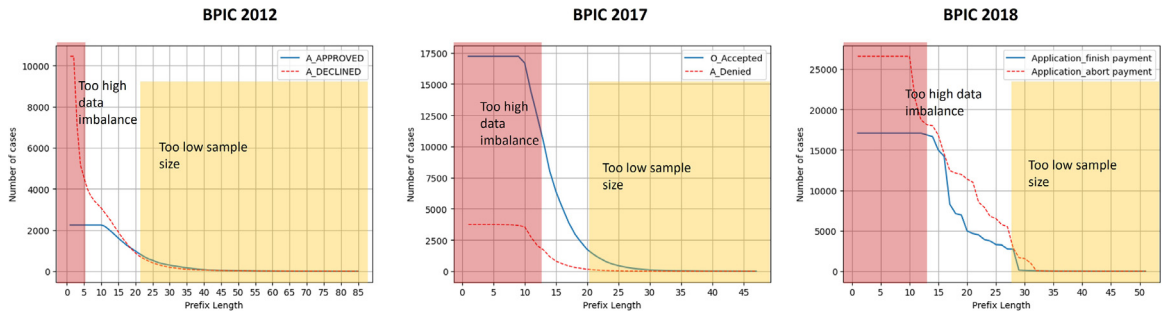


Fig. 3. Number of cases with each outcome by the prefix length.

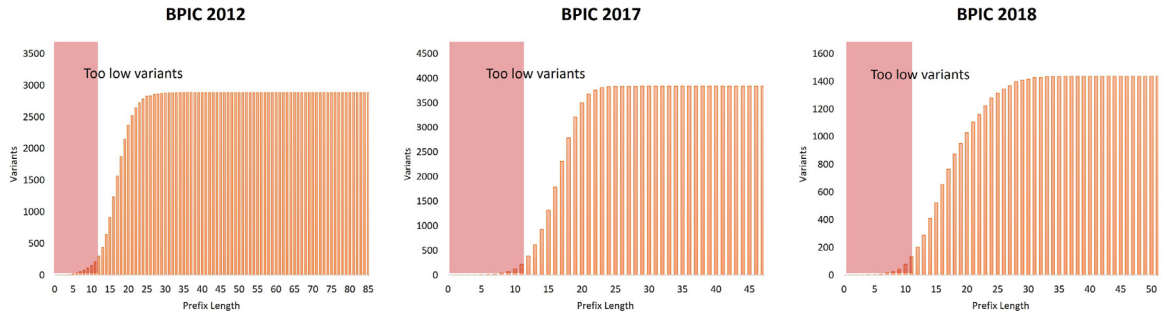


Fig. 4. Number process variants by the prefix length.

prefix. For the time elapsed feature, we used the maximum value representing the overall time elapsed in the process at the point of prediction and the rest of the numerical features were aggregated by mean representing the average value of that particular feature (e.g.: Average task duration). Therefore, this part of the case-level feature vector represents an aggregation encoded feature vector (Teinemaa et al., 2019). Next, we construct the case-specific feature vector with the case-specific attributes present in the log (e.g., requested loan amount in BPIC 2012/2017 and requested grant amount in BPIC 2018). All categorical case-specific attributes were one-hot-encoded to convert into numerical features and all numerical features were min–max normalized. The constructed feature vector is a one-dimensional vector for a given case identifier.

Then to arrive at the final case-level feature vector, the case-specific feature vector was appended to the aggregated event-level feature vector. This will facilitate the final explanation to include the contribution of those case-specific attributes as well as event-specific attributes towards the decision.

4.5. Model development and training

We deploy the logistic regression model, directly from the scikit-learn library (Pedregosa et al., 2011). The deep neural networks were developed using Tensorflow (Abadi et al., 2015) and Keras (Chollet et al., 2015) libraries and hyperparameter optimized for each dataset, using KerasTuner (O'Malley et al., 2019) library. The logistic regression model was optimized for the best possible performance by changing the maximum number of iterations. Prefixes that are generated at each prediction point are bucketed together and for each bucket, an instance for each of the three models is constructed and trained separately.

4.6. Evaluation criteria and metrics

There are two criteria used to evaluate a model explanation: fidelity and interpretability, as noted in Zhou et al. (2021). Fidelity pertains to the faithfulness of the explanation in relation to the model. Since our approach extracts explanations intrinsically from the model, fidelity is not assessed. Rather, we focus on evaluating the interpretability of our explanations, which refers to the ease with which they can be understood by people. Additionally, it is crucial for intrinsically explainable models to maintain a reasonable level of predictive accuracy, ensuring that the design of the explainability features does not adversely affect the model's predictive performance. Thus, we evaluate our models using three metrics.

4.6.1. Evaluation of accuracy

AUC-ROC stands for "Area Under the Receiver Operating Characteristic Curve". It is a performance metric used to evaluate the performance of binary classification models. The Receiver Operating Characteristic (ROC) curve is a plot of the true positive rate (TPR) against the false positive rate (FPR) for different threshold values of a binary classifier's prediction probability. The AUC-ROC is the area under this curve, which provides a measure of the classifier's ability to discriminate between positive and negative classes. The AUC-ROC ranges from 0 to 1, where a score of 0.5 indicates a classifier that performs no better than random chance, and a score of 1 indicates a perfect classifier. We measure the accuracy of the predictions made by each model with AUC-ROC, instead of using the traditional accuracy measure, which measures the classification accuracy based on a single threshold.

4.6.2. Evaluation of interpretability of explanations

In this study, we assess the quality of explanations by contrasting the global case-level explanations generated by the logistic regression model with the globally aggregated case-level explanations produced by XD^2-net. Our methodology leverages the logistic regression model to generate a case-level global explanation while using XD^2-net to generate case-level local explanations. However, as XD^2-net generates local explanations for individual instances, these explanations can be

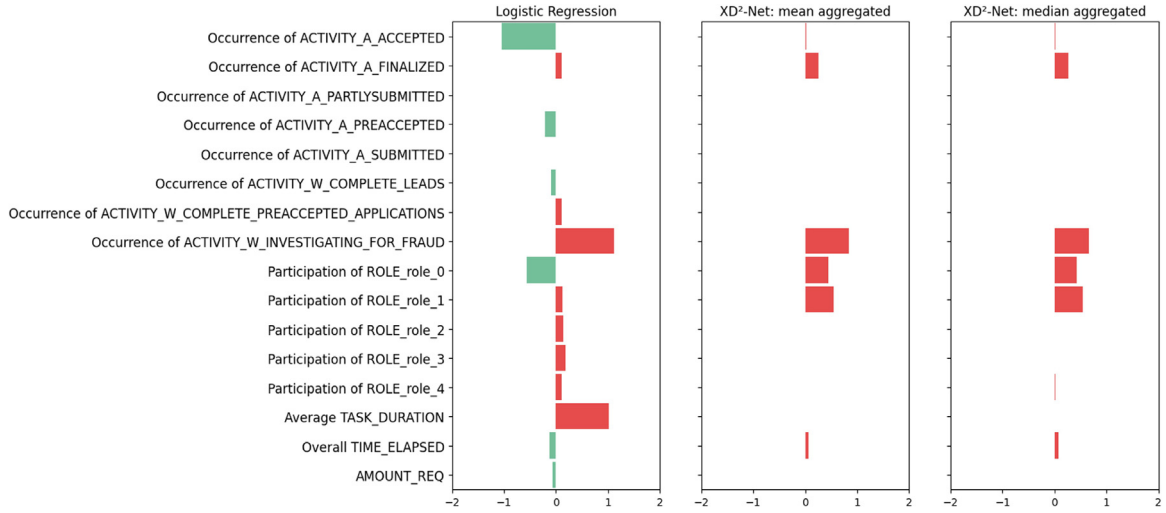


Fig. 5. Global explanation generated by logistic regression vs. the globally aggregated explanation of XD^2-net - prediction point 5.

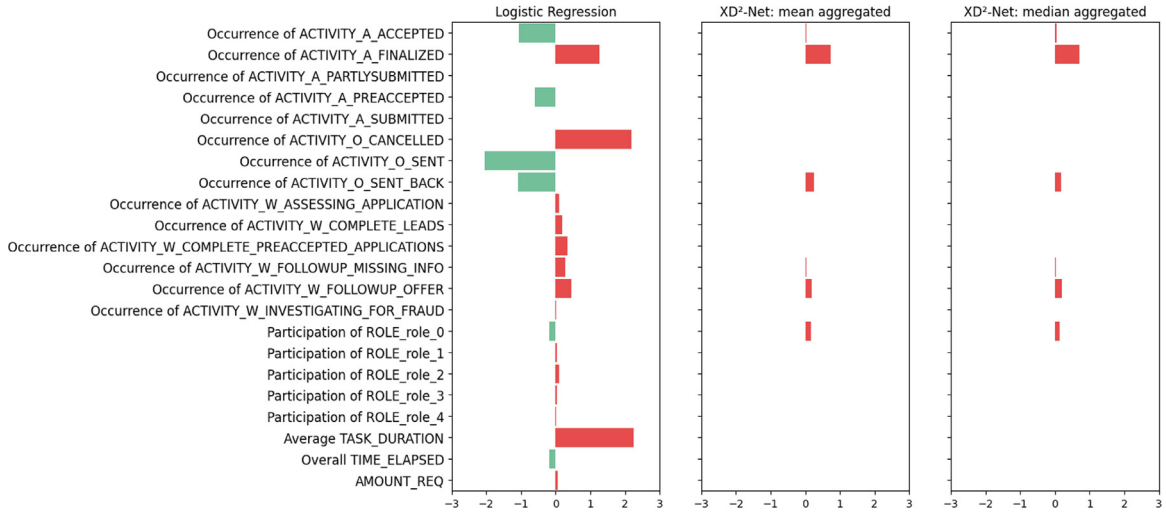


Fig. 6. Global explanation generated by logistic regression vs. the globally aggregated explanation of XD^2-net - prediction point 15.

aggregated to produce cohort or global explanations that offer a general explanation for either a selected cohort of instances or all instances. Figs. 5 and 6 depict the comparison of the global explanations generated by logistic regression and XD^2-net for prediction points 5 and 15 for BPIC 2012 respectively. Both explanations employ a feature importance-based explanation that is similar in terms of their feature space.

To compare the quality of explanations, we evaluate the globally aggregated local case-level explanations created by XD^2-net against the real global case-level explanation produced by the logistic regression model. To globally aggregate the XD^2-net case-level explanations, we utilize two aggregation methods: mean and median (of the feature importance values). We then compare each of these globally aggregated explanations to the global explanation generated by the logistic regression model to assess how well they function as global explanations that can elucidate all the predictions.

We employ two functionally grounded (computational) metrics, namely *agreement* and *effective complexity*, to evaluate the interpretability of the explanations (Zhou et al., 2021). These metrics assess how well the explanations could be understood by humans. The *agreement* measure evaluates the extent to which two explanations agree on the top features that explain a prediction (Krishna et al., 2022). It is based on the observation that people usually focus on the top k features when understanding an explanation rather than considering all the features. The *effective complexity* measure assesses the minimum number of features needed to explain a model decision in a way that does not affect the prediction if only those features are used for the prediction task (Nguyen and Martínez, 2020). If an explanation has low effective complexity, it can reduce the cognitive load on the human trying to understand it, thus enhancing the interpretability of the explanations (Abdul et al., 2020). However, we acknowledge that the interpretability of an explanation ultimately depends on how well a human user trusts and understands it Lopes et al. (2022), and these computational metrics can provide only an initial insight into how human-friendly the explanations are.

Explanation agreement: The feature agreement between the globally aggregated XD^2-net explanations and the global explanation generated

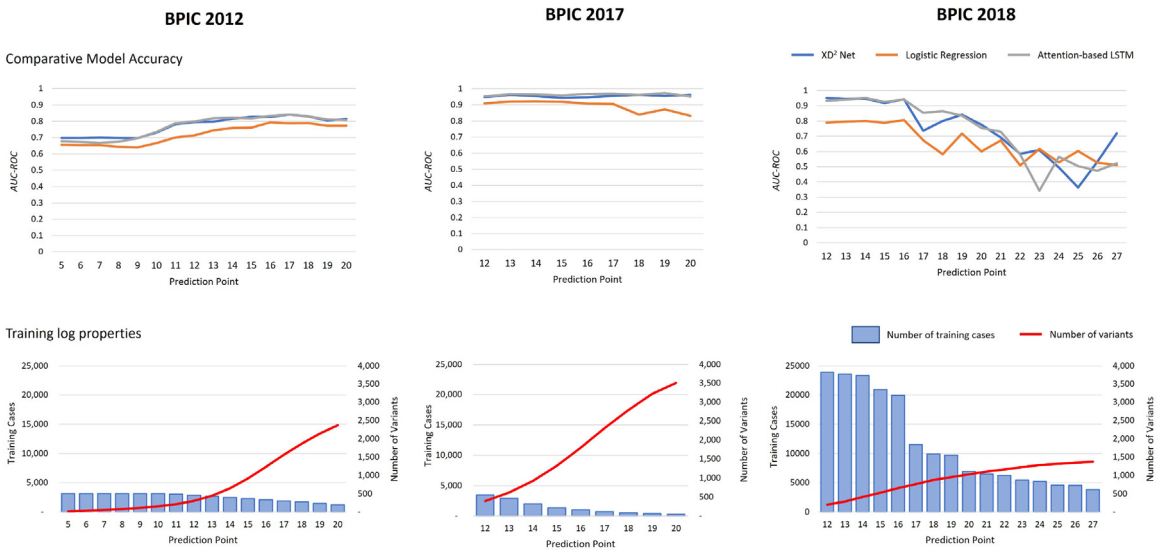


Fig. 7. Model accuracy: XD^2-net, Logistic regression and attention-based LSTM.

by the Logistic Regression is evaluated using two metrics proposed in Krishna et al. (2022). The agreement between the two explanations is evaluated based on the top 5 features identified by each of them.

- *Feature agreement* (of top k features) measures the agreement of features within top k features identified by each explanation. Given two explanations E_a and E_b, the feature agreement metric can be formally defined as:

$$Feature\_agreement(E_a, E_b, k) = \frac{|top\_features(E_a, k) \cap top\_features(E_b, k)|}{k}$$

- *Rank agreement* (of top k features) measures the agreement of features and their respective ranks within the top k features identified by each explanation. Given two explanations E_a and E_b, the rank agreement metric can be formally defined as:

$$Rank\_agreement(E_a, E_b, k) = \frac{|\cup \{s | s \in top\_features(E_a, k) \wedge s \in top\_features(E_b, k) \wedge rank(E_a, s) = rank(E_b, s)\}|}{k}$$

Effective complexity: To evaluate the effective complexity of the explanation, we evaluate how many (most significant) features by minimum will be required by the model to predict the outcome at the same level of prediction confidence as the original prediction. Mathematically, this metric is defined as follows Nguyen and Martínez (2020); Let $a^{(i)}$ be the attributions ordered increasingly w.r.t. their absolute value, and $x^{(i)}$ the corresponding features. Let $M_k = x_{N-k}, \ldots, x_N$ be the set of top k features. Given a chosen tolerance $\epsilon > 0$, the *effective complexity* is defined as

$$k^* = \underset{k \in 1,\ldots,N}{argmin} |M_k| s.t. E(l(y^*, f_{-M_k})|x^*_{M_k}) < \epsilon$$

The original formula is designed to evaluate local-level explanations using feature perturbation as the experimentation mechanism. The formula is defined to find the minimum number of important features denoted by k where, the set of minimum important features required to explain the prediction, M_k, the original prediction of the model with all features present denoted by y^*, the prediction function with the M_k important features fixed (to the original values) and the non-important features perturbed is denoted by f_{-M_k}, the error function between y^* and the result of f_{-M_k} denoted by l, and the expected value of the error function between y^* and the result of f_{-M_k} for all possible perturbations of the non-important feature set denoted by E. The objective is to determine the minimum value of k such that $E(l(y^*, f_{-M_k})|x^*_{M_k})$ is less than a pre-determined threshold ϵ, removing the maximum number of non-important features, $N - k$.

To adapt this formula for global-level explanations, we conduct an ablation. The features of the global explanation are arranged based on their importance, and a feature ablation study is performed by removing the features one by one from least to most important, re-training each model, and measuring the deviation of the retrained model's prediction confidence (probability) for a given instance from the original prediction confidence for the same instance using Root Mean Squared Error (RMSE) as the metric. The effective complexity of the explanation is then measured as the minimum number of features required to maintain the RMSE below a certain threshold ϵ compared to when computed with the total number of features.

4.7. Results

This section provides a detailed analysis of the results obtained from the experiments, which includes the accuracy of the model predictions, along with the explanation agreement and the effective complexity of the global explanations. Furthermore, the results section includes an in-depth interpretation of the observed patterns and relationships between the performance metrics and the nature of the datasets and explanations.

4.7.1. Model accuracy

The accuracy of outcome prediction by the three evaluated models is depicted in Fig. 7, along with the properties of the training log that can aid in the interpretation of performance. In the best case, we expect the deep learning-based models will outperform the simple and transparent logistic regression model, and the acceptable worst case will be for all three models to display a similar level of accuracy. In the results, we observe that the model's performance improves when there is an increase in either or both of the following factors: (1) the number of available data points for training (i.e., the number of cases), and (2) the diversity of information (i.e., the number of process variants). Conversely, we also notice a decrease in model performance when there is a low number of cases or variants. Another pattern we notice is deep learning-based architectures (XD^2-net and attention-based LSTM) tend to perform better with a high number of training samples, even if there are fewer variants (BPIC 2018 prediction points 12 to 16 versus BPIC 2012/BPIC 2017). In the experiments, all three models exhibit comparable performance in the worst case, but if provided with a significantly larger dataset for training, the deep learning-based models

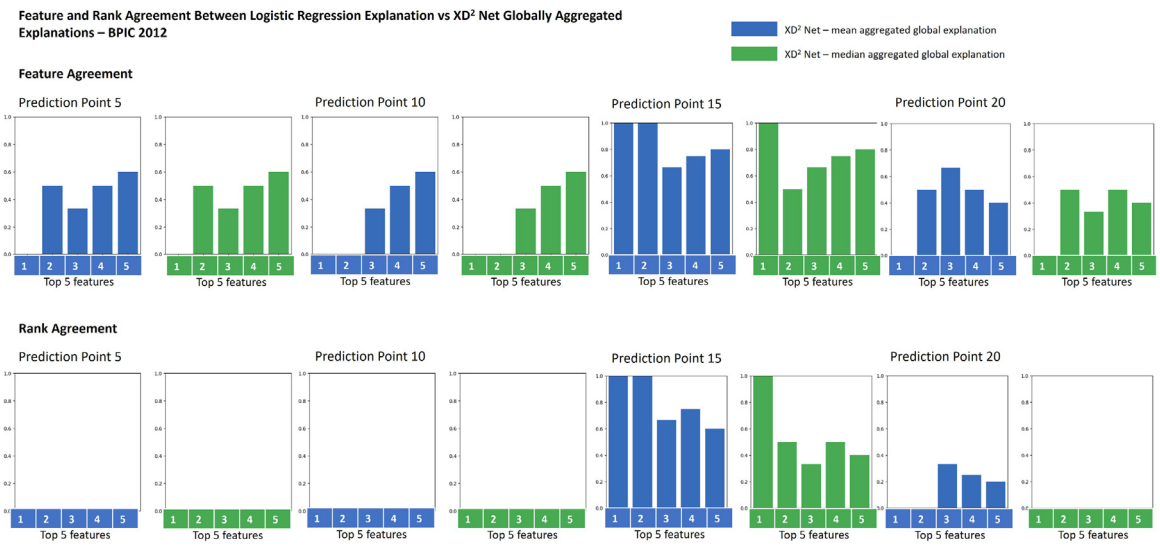


Fig. 8. Explanation agreement for BPIC 2012: For prediction points 5,10 and 20 there is a weak feature agreement between the two explanations with none of the explanations agreeing in terms of how the top 5 features are ranked, whereas for prediction point 15, the two explanations agree with each other much better.

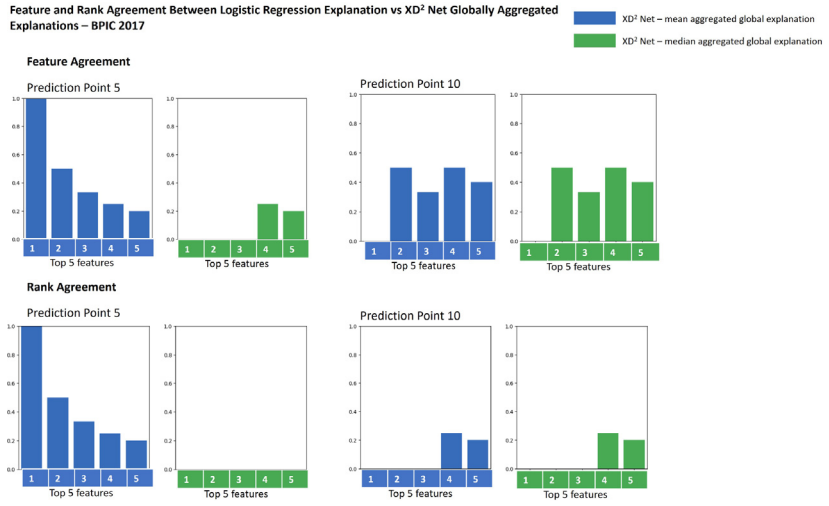


Fig. 9. Explanation agreement for BPIC 2017: For prediction point 15, XD^2-net mean aggregated global explanation agrees with logistic regression explanation for the top one feature, whereas for prediction point 20, both XD^2-net global explanations agree with logistic regression explanation for the top fourth feature.

outperform the transparent logistic regression model (the best case), in addition to their ability to generate local explanations that can explain a single prediction. Depending on specific requirements in terms of explainability and performance, one or more of these architectures can be chosen based on the nature of the dataset.

4.7.2. Explanation agreement

In explanation agreement, we evaluate to what extent would the two globally aggregated explanations from XD^2-net agree with the global explanation extracted from the logistic regression. We compare the top 5 important features (based on the feature importance value) for XD^2-net mean aggregated global explanation and XD^2-net median aggregated global explanation against the logistic regression global explanation, to evaluate the feature and rank agreement based on the previously defined formula.

Figs. 8, 9, and 10 depict the feature and rank agreement between XD^2-net explanations and logistic regression explanation for BPIC 2012, 2017 and 2018 logs respectively. The blue bars depict the feature and rank agreement of mean-aggregated XD^2 explanations, and the green bars depict the feature and rank agreement of median-aggregated XD^2 explanations. The x-axis of each bar graph shows the top 5 features based on the feature importance, and the y-axis shows the agreement. The experiment was done for a chosen set of prediction points that cover the total range of prediction points for which the outcome prediction was performed. An agreement value of 1 indicates that there is a perfect agreement between the top features identified by each of the explanations, whereas a value between 0 and 1 indicates that only a fraction of the top features agree with each other.

Based on the explanation agreement results for different decision points for each of the logs, we can observe that in most of the instances, the explanation agreement between the logistic regression explanation and XD^2-net mean aggregated global explanation vs. explanation agreement between logistic regression explanation and XD^2-net median aggregated global explanation do not vary significantly. For all the instances we can see that at least one feature out of the top 5 features agrees between the logistic regression and XD^2-net explanations, and in 7 out of 10 instances, one of the top 2 features show such agreement.

4.7.3. Effective complexity

In this evaluation, we observe what the effective complexity (i.e. the number of minimum most important features required to explain the

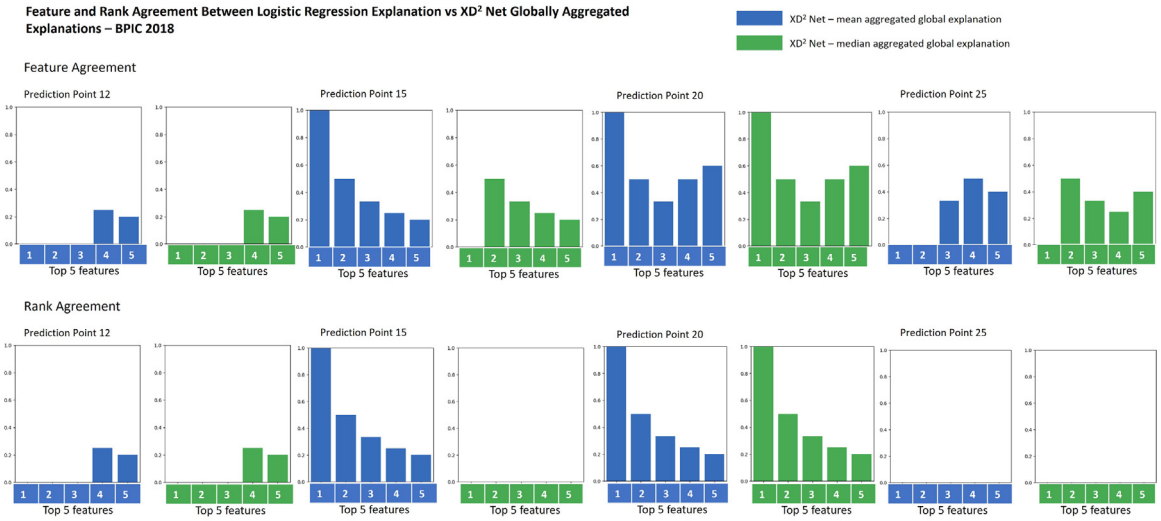


Fig. 10. Explanation agreement for BPIC 2018: For prediction point 20, the explanation agreement between the XD^2-net explanations and logistic regression explanation is better compared to the other decision points.

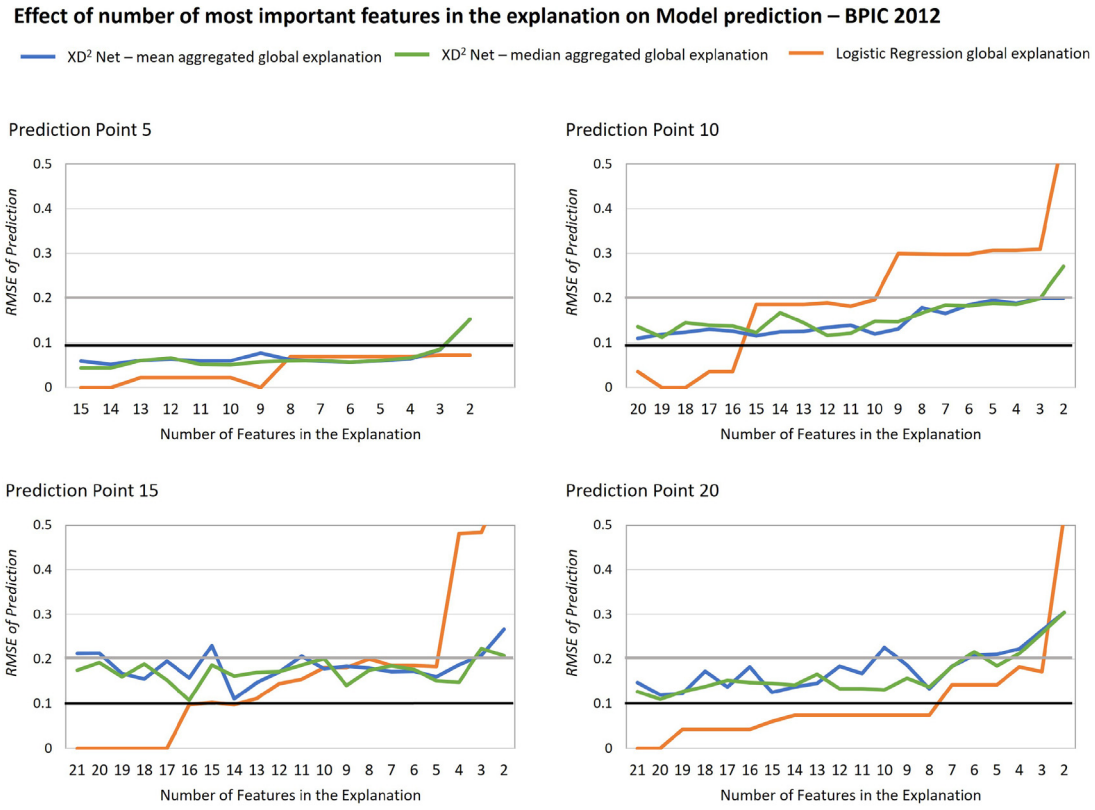


Fig. 11. Evaluation of effective complexity for BPIC 2012: Global explanation extracted from logistic regression shows a better (lower) effective complexity when $\epsilon = 0.1$, whereas the globally aggregated explanations extracted from XD^2-net show a better effective complexity for $\epsilon = 0.2$.

prediction) of the global explanations generated by the logistic regression model and the globally aggregated local explanations generated by the XD^2-net is. We assess the number of features each explanation requires to maintain an error (RMSE) between the original prediction and the revised prediction with fewer features below a specific threshold (ϵ). We evaluate what the effective complexity for $\epsilon = 0.1$ and $\epsilon = 0.2$ is. Provided that a model prediction is the probability of the process outcome which is a number between 0 and 1, we do not perform the evaluation beyond the point where RMSE = 0.5, which indicates a serious error in the prediction. We perform this comparison on all three datasets and a chosen set of representative prediction points that cover the entire range of prediction points utilized to predict the process outcome. Figs. 11, 12, and 13 depict the change of RMSE of the revised model prediction when each of the explainable models is retrained and retested with least important features being removed one by one.

In BPIC 2012 log-related experiments (Fig. 11), we can see that at prediction point 5, all three explanations can go up to an effective complexity of 2 minimum features whilst maintaining an error between the original prediction and the prediction with top 2 features (RMSE) $< \epsilon = 0.1$. However, for prediction points 10, 15 and 20 the globally aggregated XD^2-net local explanations cannot reduce any features from the original explanation if the $\epsilon = 0.1$ (black horizontal line in the

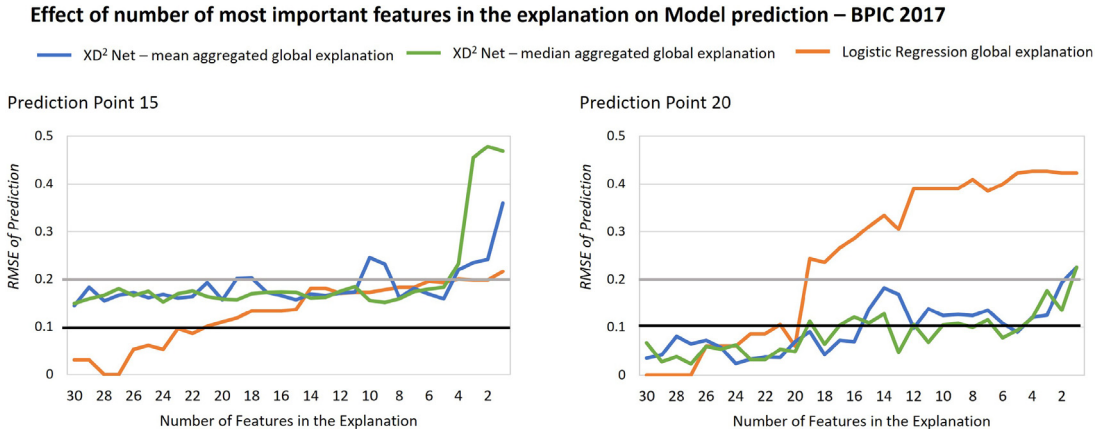


Fig. 12. Evaluation of effective complexity for BPIC 2017: For prediction point 15, Global explanation extracted from logistic regression shows a better effective complexity for both thresholds of ϵ, whereas, for prediction point 20, the globally aggregated explanations extracted from XD^2-net show better effective complexity for both thresholds.

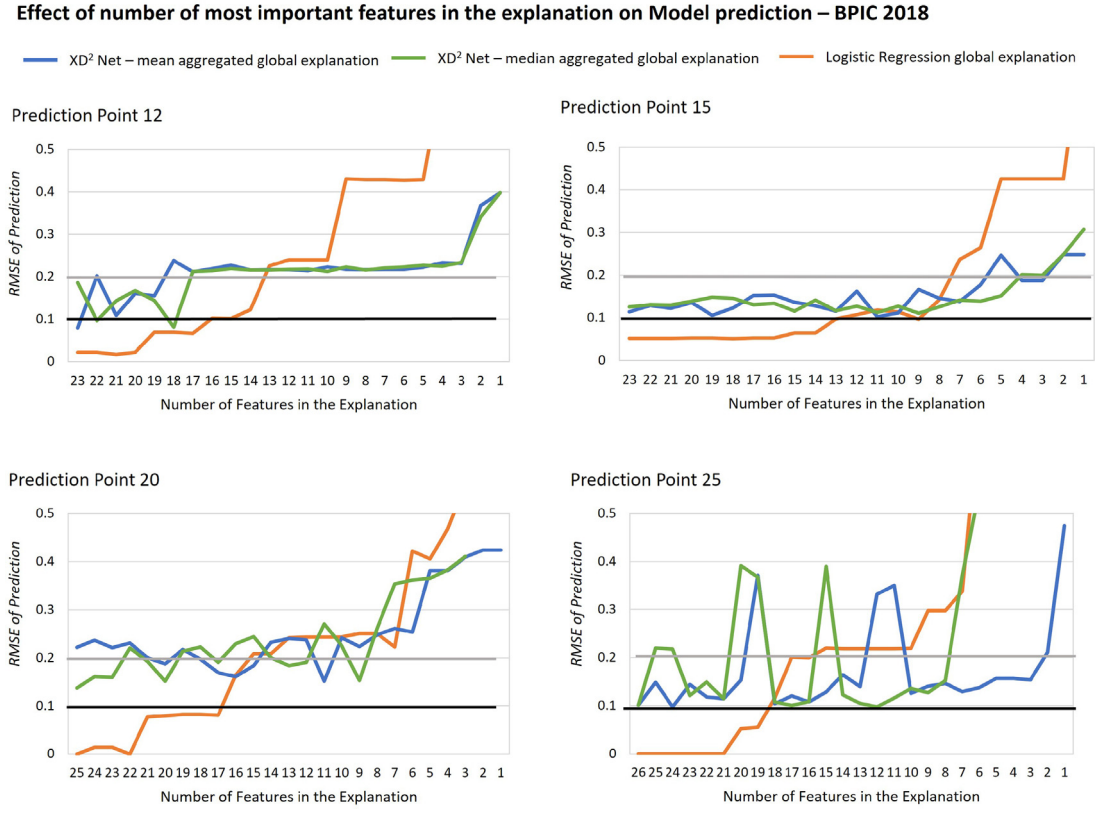


Fig. 13. Evaluation of effective complexity for BPIC 2018: Except for prediction point 15, Global explanation extracted from logistic regression shows a better effective complexity for both thresholds of ϵ.

graphs), whereas for the global explanation extracted from logistic regression can reach an effective complexity of 16 (out of 20) features for the prediction point 10, 16 (out of 21) features for the prediction point 15 and 7 (out of 21) features for the prediction point 20. For a more relaxed $\epsilon = 0.2$ (grey horizontal line in the graphs), the effective complexity for globally aggregated XD^2-net local explanations can go up to 3 features for prediction points 10 and 15, and 4 features for prediction point 20. For BPIC 2017 and 2018 logs also for different prediction points, we observe different results for effective complexity between logistic regression and XD^2-net global explanations, where some instances XD^2-net shows better effective complexity in global explanations (prediction point 20 in BPIC 2017 and prediction point 15 in BPIC 2015), whereas in other instances global explanation extracted from logistic regression model performs better.

However, a notable pattern is the rate of change in prediction error (RMSE) when the least important features are removed progressively

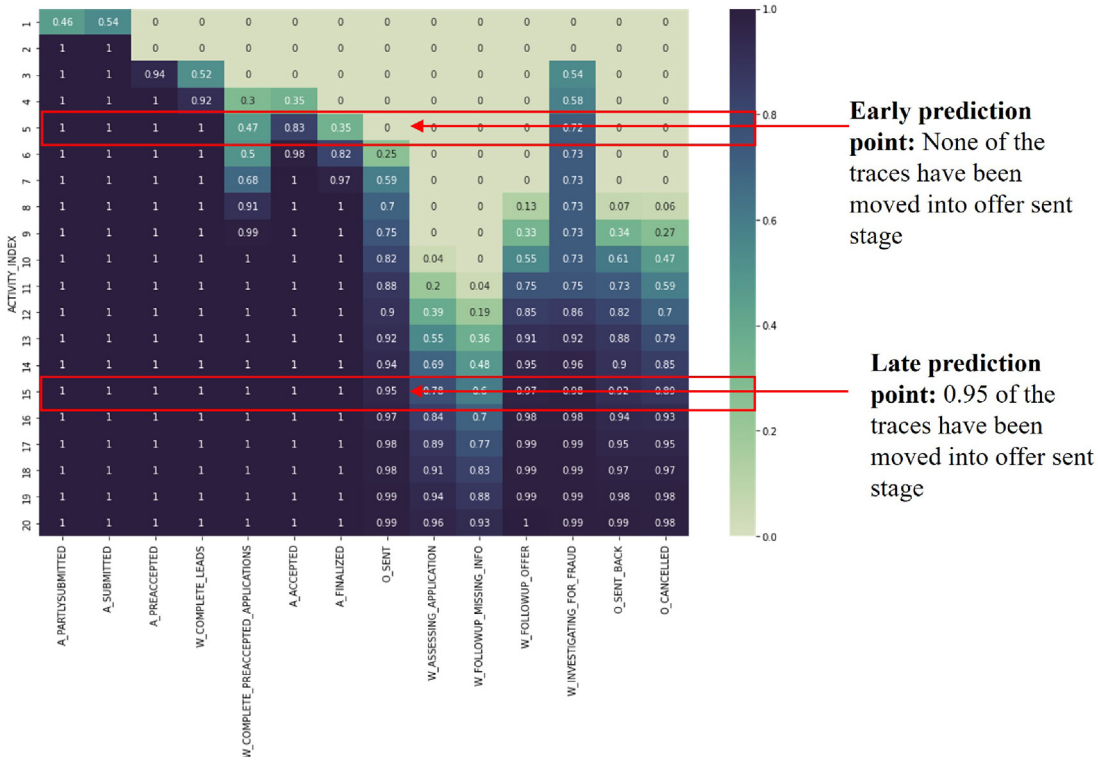


Fig. 14. The two prediction points are chosen on the basis of how many applications have been moved to the loan offer sent (O_SENT) stage.

is less for globally aggregated XD^2-net local explanations, compared to the inherently global explanation extracted from the logistic regression. This effect is most visible in prediction point 15 of BPIC 2012 and prediction point 25 of BPIC 2018. This could potentially be due to the fact that globally aggregated XD^2-net local explanations not being able to represent true global explanations that are applicable to all predictions. As a result, removing a feature based on its aggregated importance may affect the prediction of different data points (i.e. process prefixes) in distinct ways.

5. Analysis of explanations

In this section, we analyse the multi-level explanations generated to answer 'how' and 'why' a particular prediction was made for the BPIC 2012 log. BPIC 2012 represents a process of loan application in a Dutch financial institution (van Dongen, 2012). It has two key outcomes, the loan application getting a successful outcome (A_SUCCESSFUL) or an unsuccessful outcome (A_UNSUCCESSFUL). We choose this event log to demonstrate the explanation as it is an easily understood process that does not require highly specific domain knowledge.

What are the different levels of explanations to demonstrate: Global explanation with case level feature attribution gives a high-level understanding of how the model makes the prediction, which is generalizable for all the samples. A global explanation with case-level attributes is directly generated by the logistic regression model, as well as globally aggregating the local explanations generated by XD^2-net. *Local explanation with case level feature attribution* gives an understanding of why a particular decision was given to a particular trace, using aggregated case-level features. We can generate these explanations with XD^2-net. If a process analyst wants to investigate further the reason why a certain decision was made upon a certain process trace, he can use *Local explanation with event level feature attribution*, which tells him how specific events and those event-specific attributes influenced the model decision. These explanations can be generated using the attention-based LSTM mechanism. With the multiple levels of explanations, we primarily try to answer the following questions.

- **How is the decision made if a loan application gets unsuccessful in general?** - This question is answered by the case-level global explanations generated by the logistic regression model
- **Why did a particular application get unsuccessful or successful?** - This question is answered by the local explanations generated by XD^2-net, which gives a specific explanation for the particular application with case-level features. The event-level explanations that are extracted from attention weights of the LSTM network help to explore this explanation further.
- **What is the difference between two similar loan applications (in terms of the requested loan amount) which got opposite outcome predictions?** - Comparative analysis of local (case-level and event-level) explanations.

It is crucial to highlight that despite the three models generating explanations to elucidate the same phenomena, they are trained independently, leading to potential differences among their explanations and resulting in non-alignment in explaining the same prediction. For instance, when comparing the global case-level explanation for a specific outcome (i.e., loan application getting unsuccessful in our demonstration) with the local case-level explanation for a particular case with the same outcome prediction, there may be inconsistencies. In the subsequent demonstration, we illustrate how each of the three questions is addressed by the respective explanations.

We demonstrate these three levels of explanations at two prediction points, one very early in the process (prediction point 5) and another very late in the process (prediction point 15) to depict how differently the model makes decisions at each prediction point. We also support our explanation through the relevant process graphs and exploratory data analysis to evaluate the validity/further strengthen the message conveyed by the explanations. As per Fig. 14, prediction point 5 occurs

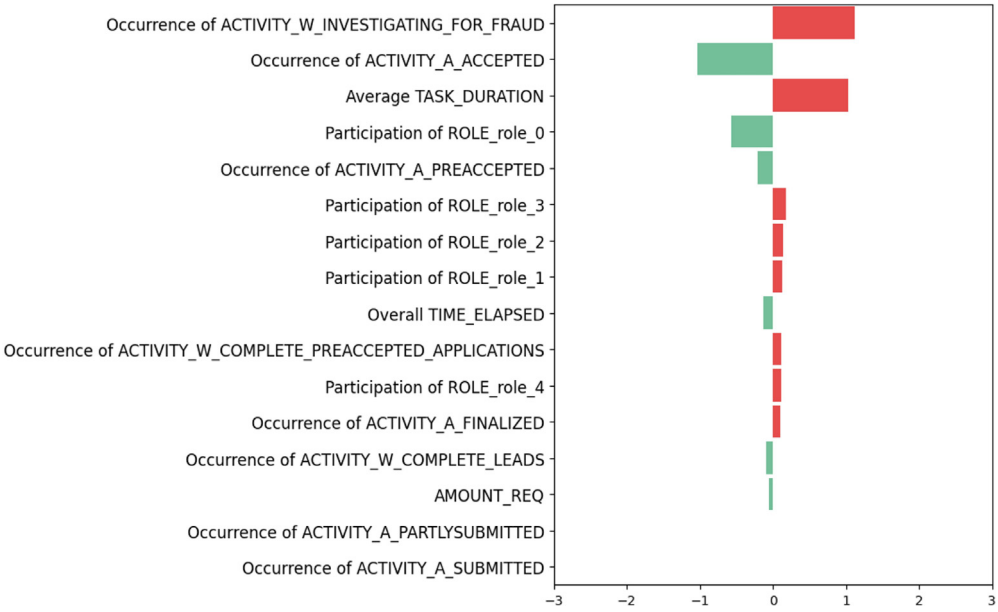


Fig. 15. Global explanation generated by logistic regression model identifies the activity W_INVESTIGATING_FOR_FRAUD and Average TASK_DURATION to be the top features that influence the decision of A_UNSUCCESSFUL.

much early in the process, and not many traces have progressed too far in the process, with none of the traces having moved to the loan offer stage. In fact, the applications at prediction point 5 represent those that are accepted and evaluated for completeness. As a result, the predictive model may not make a very good prediction due to the insufficiency of the information. For such a model, the model explanations could serve as a debugging tool that can help the data-scientists can understand the reason for poor model performance (Wickramanayake et al., 2022b)

Prediction point 15 is an advanced stage of the application process, and the model has a rich set of information to make the decision with, where 95% of the eligible traces have been moved past the loan offer sent stage. At this stage, model explanations can help the process scientists to understand the process attributes that lead towards a desired outcome (Mehdiyev and Fettke, 2021).

5.1. Explaining outcome predictions at an early stage of the process

At prediction point 5, our answers to the three questions stipulated above are based on the explanations that are generated with the limited amount of information that is available for the three models, given the process is at an early stage by this point. While these explanations may not offer compelling reasons for business users to understand the reasoning behind specific outcome predictions, they can be valuable for data scientists to enhance the performance of the model itself (see Fig. 15).

Case level - Global explanations: How does the model make the decision if an application gets an unsuccessful outcome in general? The global explanation with logistic regression reveals that the occurrence of the activity W_INVESTIGATING_FOR_FRAUD (once or multiple times) and a high average TASK_DURATION are key features that significantly impact the likelihood of an application being unsuccessful. On the other hand, the occurrence of the A_ACCEPTED activity and the involvement of Role_0 are influential factors towards an application being successful. From a business perspective, these observations align with expectations, making logical sense in terms of their impact on the decision-making process. A high average event duration means the bank is taking more time to process the application, likely due to the application having issues. W_INVESTIGATING_FOR_FRAUD has a very high bearing on the decision, as 100% of such applications do get unsuccessful, however, the number of such applications is not substantial.

Case level and event level - Local explanations: Why did the application 200 775 get unsuccessful? Fig. 16 depicts the local explanation for the application (case ID) 200775, which is an unsuccessful application. This explanation consists of the local case-level explanation generated by XD^2-net along with the prediction confidence of XD^2-net (top right), the local event-level explanation generated by attention-based LSTM model along with the prediction confidence (bottom right) and the process model that depicts the process path taken by the entire trace with the part that the predictive model cannot see (due to prefix truncation) greyed out (left). In this explanation, local case-level and local event-level explanations tell which features influenced the model prediction (primary explanations), and the process model helps to validate and support the explanation.

For this specific application which got an unsuccessful outcome prediction, the case-level local explanation suggests the occurrence of the activity W_COMPLETE LEADS thrice is the main contributing factor. As per the process model, we can observe that this is an application that got declined early in the process without even being subjected to a successful loan offer. It has gone through W_COMPLETE LEADS three times. The event-level explanation identifies three equally important features for the decision, W_COMPLETE_LEADS at event 5 (final event before prediction), TIME_ELAPSED at event 5 to be 0.4 days and the involvement of role_1 and event 5. Overall, the event-level explanation identifies the last event of the prefix to be the most influential towards the decision.

Case level and event level - Local explanations: Why did the application 189466 get successful? This specific process trace appears to be exceptionally efficient, as per the process map, indicating minimal issues with the application. Unlike the previous instance, where the first 5 events took nearly half a day to execute (which is more than a business day in general), in this case, they occurred within an hour,

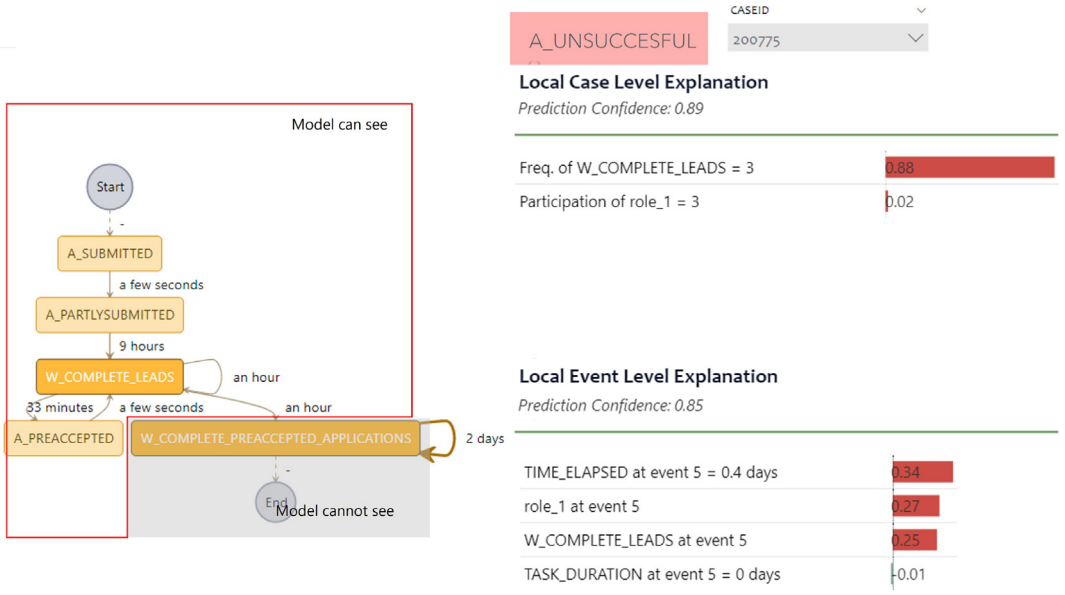


Fig. 16. Local explanations suggest three occurrences of W_COMPLETE_LEADS activity influence the prediction of A_UNSUCCESSFUL outcome for the loan application 200 775 with a 0.9 confidence.

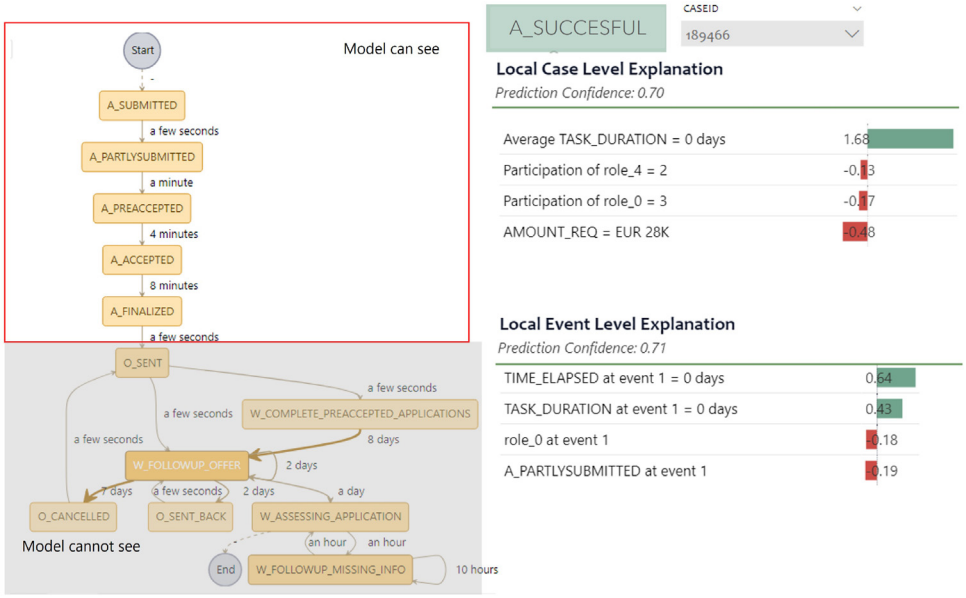


Fig. 17. Local explanations suggest the low average TASK_DURATION influences the prediction of A_SUCCESSFUL outcome for the loan application 189 466.

suggesting a smooth process flow. The case-level explanation further reveals that a low average TASK_DURATION positively impacts the decision for loan application approval, while a high loan amount requested (EUR 32,000) has a negative influence on the decision. However, the local event-level explanation seems to rely solely on the first event of the trace, which may not provide substantial insights into the prediction decision (see Fig. 17).

What is the difference between two similar loan applications (in terms of the requested loan amount) which got opposite outcome predictions? Case IDs 185461 and 200775 both correspond to loan applications with loan request amounts ranging from 5000 to 6000 Euros, considered as low loan request amounts. In the case where the outcome prediction is A_SUCCESSFUL, the case-level explanation highlights that the low loan amount requested (EUR 6K) and the short average event duration (0 days) are contributing factors to the prediction. Conversely, for case ID 200775, the explanation suggests that the occurrence of the W_COMPLETE_LEADS activity three times has influenced the model's prediction of A_UNSUCCESSFUL.

At the event level, the explanation for case ID 185561 focuses primarily on the very first event, while for case ID 200775, event 5, during which the W_COMPLETE_LEADS activity occurs, is found to

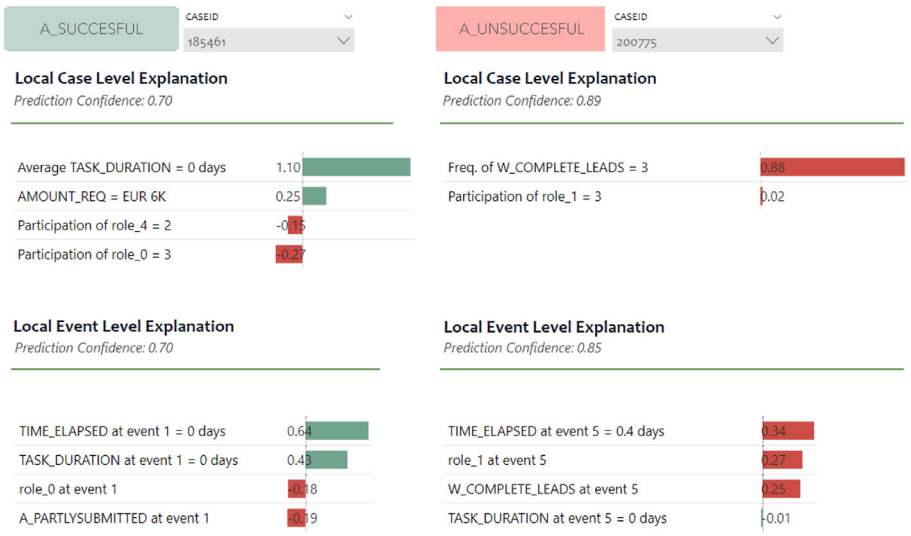


Fig. 18. Low average TASK_DURATION and low Loan amount requested influence the prediction of outcome A_SUCCESSFUL for Case ID 185461 whereas the occurrence of W_COMPLETE_LEADS three times influences the prediction of A_UNSUCCESSFUL for Case ID 200775, despite both applications being of similar loan amounts (EUR 6000).

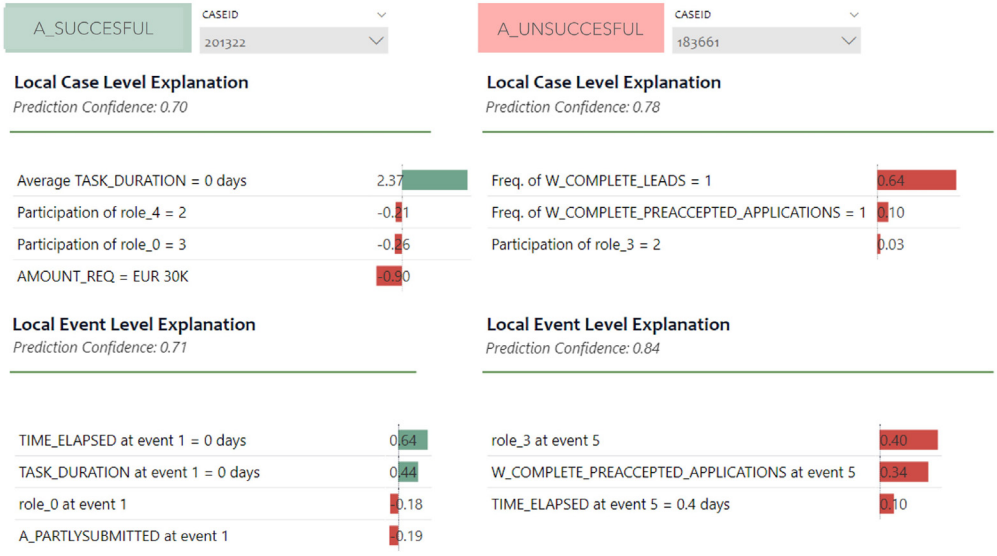


Fig. 19. Low average TASK_DURATION influences the prediction of outcome A_SUCCESSFUL for Case ID 201332 whereas the occurrence of W_COMPLETE_LEADS once influences the prediction of A_UNSUCCESSFUL for Case ID 183661, despite both applications being of similar loan amounts (EUR 20,000).

be most influential in the decision for the application to receive an unsuccessful outcome prediction (see Fig. 18).

When comparing two loan applications with loan amounts requested exceeding EUR 20,000, and having opposite outcome predictions (as shown in Fig. 19), the explanations exhibit similarities to the previous comparison. However, in this case, the high loan amount requested is observed to have a negative impact towards the A_SUCCESSFUL outcome prediction.

5.2. Explaining outcome predictions at a later stage of the process

Given the availability of additional information at prediction point 15, it is anticipated that the model explanations would be more insightful and informative.

Case level - Global explanations: How does the model make the decision if an application gets unsuccessful in general? As per the global explanation derived from logistic regression, the features that consistently hold high influence towards an application being unsuccessful are the occurrence of the O_CANCELLED activity (indicating cancellation of the loan offer) and a high Average TASK_DURATION. On the other hand, the features that significantly impact a successful outcome for the application are the occurrence of the O_SENT activity (indicating a loan offer has been sent to the customer) and the occurrence of the O_SENT_BACK activity (indicating customer acceptance of the loan offer).

Case level and event level - Local explanations: Why did the application 202596 get unsuccessful? In Fig. 21, an application that received an unsuccessful outcome prediction due to the loan offer being cancelled after 7 follow-up attempts, with a total process duration of 11 days. The case-level explanation attributes the most influential feature for the prediction to be the occurrence of the W_FOLLOWUP_OFFER activity seven times. However, the event-level explanation differs slightly, indicating that the total time taken by the process up to the prediction point (event 15) has the highest influence, along with the last occurrence of the W_FOLLOWUP_OFFER activity and its associated role.

While both explanations agree that the W_FOLLOWUP_OFFER activity influences the A_UNSUCCESSFUL prediction, the event-level explanation specifically identifies the last occurrence of this activity as

Fig. 20. Global explanation generated by logistic regression model identifies the activity O_CANCELLED and high Average TASK_DURATION to be the top features that influence the decision of A_UNSUCCESSFUL.

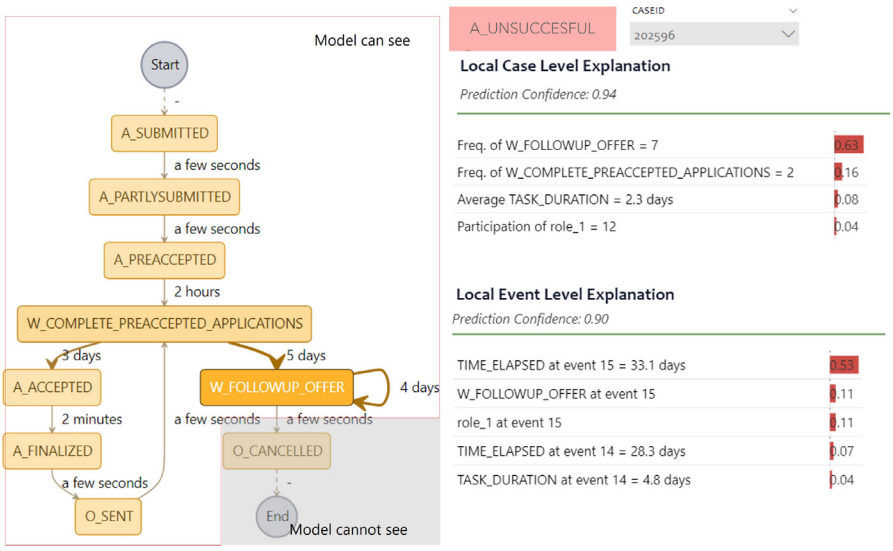


Fig. 21. Local explanations suggest the occurrence of W_FOLLOWUP_OFFER activity to influence the prediction of A_UNSUCCESSFUL outcome.

the most influential. Notably, this explanation contradicts the global explanation for unsuccessful loan applications (as shown in Fig. 20). The global explanation recognizes the occurrence of the W_FOLLOWUP_OFFER activity as only marginally influential towards the A_UNSUCCESSFUL prediction, and it is not listed among the most influential features.

Case level and event level - Local explanations: Why did the application 180703 get successful? The loan application underwent swift processing without any instances of O_CANCELLED activity. According to the case-level explanation, the key factors that influenced the decision of A_SUCCESSFUL outcome were the low average TASK_DURATION and the low loan amount requested. However, the occurrence of W_COMPLETE_PREACCEPTED_APPLICATIONS activity six times had a detrimental impact on the decision. In contrast, the local event-level explanation provided limited information, identifying only the first event in the trace as influential in the decision (see Fig. 22).

What is the difference between two similar loan applications (in terms of the requested loan amount) which got opposite outcome predictions? Loan applications with Case IDs 193497 and 197008 have requested loan amounts ranging from 5000 to 6000 Euros, as depicted in Fig. 23 (top half). Case ID 193497 was successful (A_SUCCESSFUL) based on the case-level explanation, which cited its low average task duration and requested loan amount as influencing factors. On the other hand, Case ID 197008 was unsuccessful (A_UNSUCCESSFUL) due

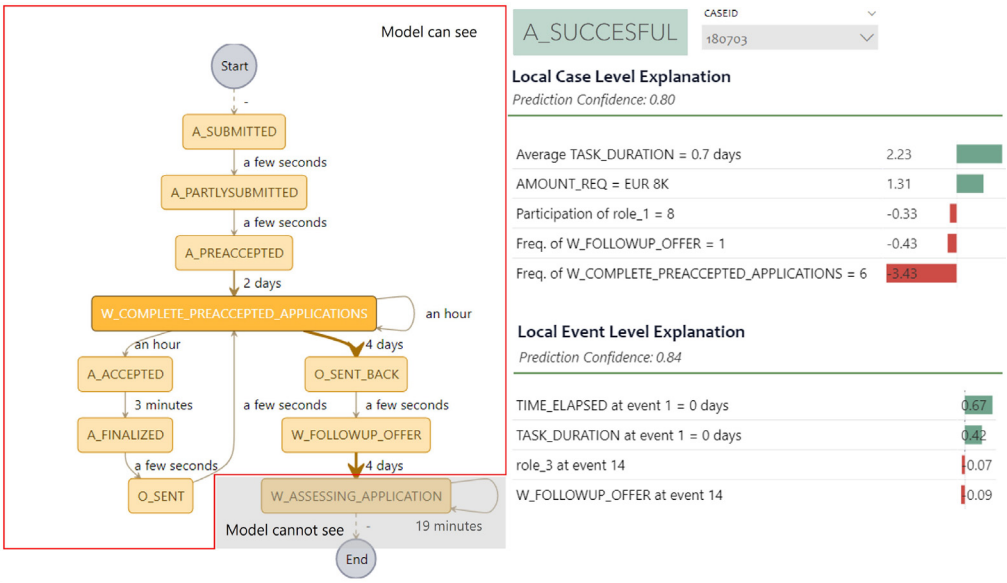


Fig. 22. Low loan amount requested and the low average TASK_DURATION are factors that influence the prediction of A_SUCCESSFUL for the loan application with ID 180703.

to the presence of W_FOLLOWUP_OFFER activity occurring five times (indicating delays in the customer acceptance of the loan offer) and W_COMPLETE_PREACCEPTED_APPLICATIONS activity occurring four times (indicating issues of the original application submitted).

The event-level explanation for the A_SUCCESSFUL prediction did not yield much insight, whereas the event-level explanation for the A_UNSUCCESSFUL prediction revealed that the most influential factor was the total time elapsed of 15.8 days at event 15. Notably, even for loan applications with higher requested loan amounts, the explanations for the outcomes were similar to those with lower loan amount requests. Thus, the explanation for the outcomes of loan applications with higher requested loan amounts appeared to be consistent with the explanation for the outcomes of applications with lower loan amount requests.

5.3. Limitations

This approach proposes three levels of explanations: a case-level global explanation generated by the logistic regression model, a case-level local explanation generated by the XD^2-net, and an event-level local explanation generated by the attention-based LSTM model. Despite explaining the same phenomena, these explanations are generated by three independent models that are fed with different feature sets and trained independently. As a result, there is a problem where these three explanations may not complement each other in explaining the same prediction, because although each individual explanation is faithful to the model from which it is extracted, they may not be faithful to each other.

Moreover, it is important to note that the explanations generated by any of these methods simply reveal how each model arrives at its decision and may not align with domain knowledge. The inherent explanations for a model's behaviour stem from its trained weights and outputs. Even if a particular feature lacks practical significance, it may still be considered important by the model if it contributes to optimizing classification accuracy. Therefore, these explanations may necessitate further interpretation, customized for the end-users, to facilitate their understanding.

6. Conclusion and future work

We have presented an approach for generating multi-level intrinsic explanations for process outcome predictions. The approach draws upon two levels of feature vectors: case-level and event-level, and three model architectures: logistic regression, attention-based LSTM, and an ensemble architecture XD^2-net. Using three publicly available datasets, we have tested the applicability of the approach as well as examined the multi-level explanations generated by the approach through an elaborate case study.

One limitation of our work is that despite introducing three levels of explanations to elucidate the same phenomena using three intrinsically explainable models, these explanations may not be able to complementarily explain the same prediction due to the independent training of the models. To address this drawback, we have devised two strategies. First, we plan to extract event-level local explanations directly from the attention-based LSTM backend of the XD^2-net itself. Second, we aim to develop an ensemble architecture that makes a voting-based final prediction to connect the XD^2-net and the logistic regression model. This approach will allow us to extract all three levels of explanations from the same ensemble model, facilitating the use of the three explanations in a complementary manner.

Secondly, in this work, we have limited our evaluation of explanation interpretability only to functionally grounded techniques. As one of our key future contributions to the research of this work part of we have devised a plan based on a framework for user-evaluation of explanations (Chromik and Schuessler, 2020) to conduct a human-oriented user evaluation to evaluate the understandability (Lopes et al., 2022) aspect of our explanations. Finally, we have limited the feature construction method of the case-level feature set to simple aggregation of event-level features over the event axis, of which the information is only limited to the overall frequency of activities and associated roles, total time taken by the process up to the prediction point and averages of other numerical features considered. In future work, we expect to enrich the explanations generated by XD^2-net by introducing techniques of generating case-level feature construction with domain-informed methods. Another direction is to develop a counterfactual

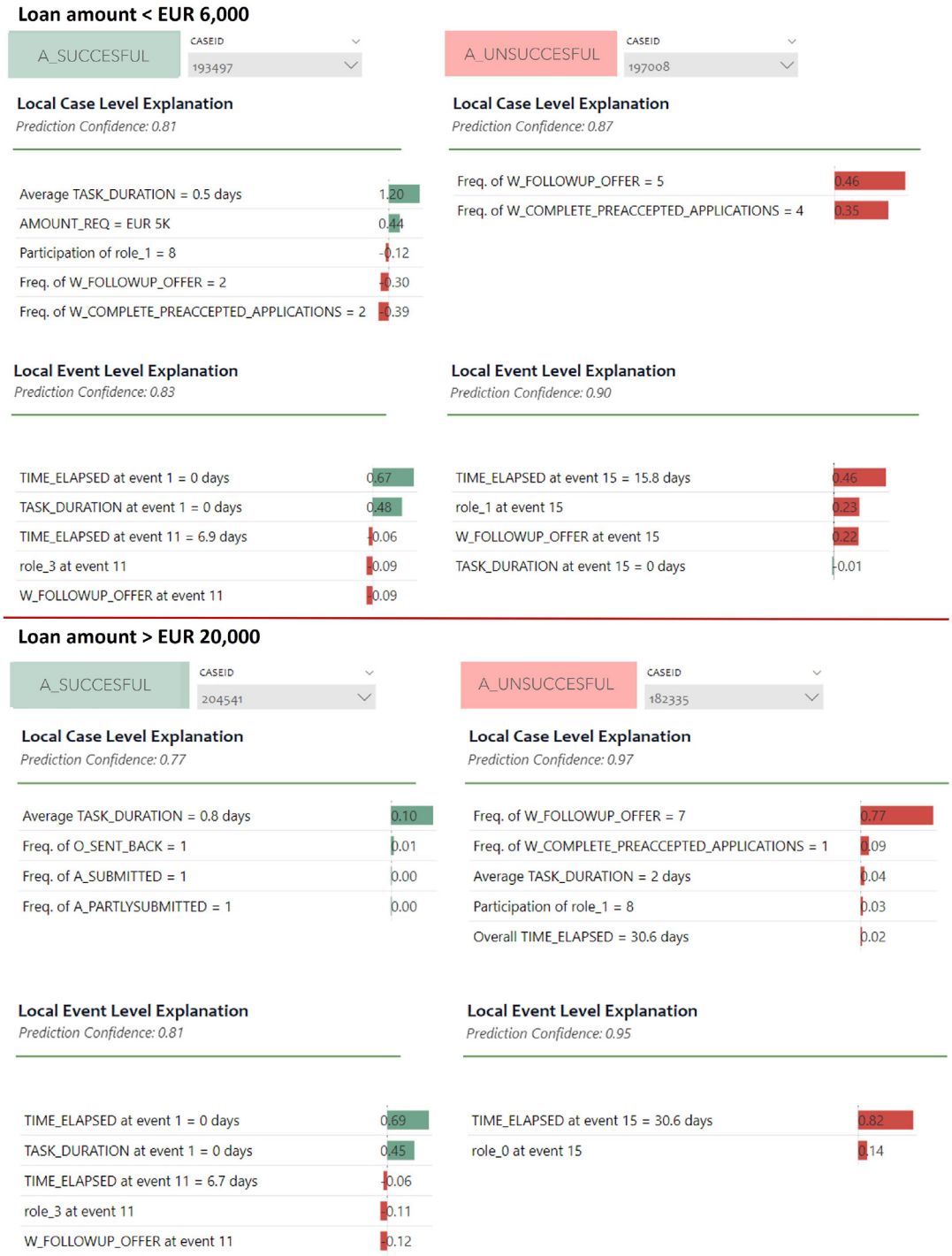


Fig. 23. Low loan amounts requested and low average TASK_DURATION influences A_SUCCESSFUL decisions whereas many occurrences of W_FOLLOWUP_OFFER and W_COMPLETE_PREACCEPTED_APPLICATIONS activities together with long process durations influence A_UNSUCCESSFUL decisions.

generation algorithm that is customized for the dual learning architecture of XD^2-net. We also expect to expand this architecture to other areas of business process prediction such as next activity prediction and process remaining time prediction. Considering the inherent capacity of XD^2-net to generate intrinsic local explanations, an intriguing avenue for further investigation involves the utilization of these explanations to gain insights into the presence of concept drift within a dataset (Demšar and Bosnić, 2018). In principle, local explanations are dedicated to explaining predictions of individual traces, which makes it possible to reveal discrepancies between predictions of different traces that could result from the presence of concept drift. This prospect has potential as a future step in the research.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

The source code for the experiments in the papers can be downloaded from the Github repository https://github.com/bemali/XDD-Net.

Acknowledgements

The reported research is part of a Ph.D. project supported by a Science and Engineering Faculty scholarship and a Centre for Data Science top up scholarship at Queensland University of Technology (QUT), Australia. This work is supported by the UNESCO Chair on AI&XR, and through the Portuguese *Fundação para a Ciência e a Tecnologia (FCT)*[http://dx.doi.org/10.13039/501100001871] under grants no. 2022.09212.PTDC (XAVIER) and no. UIDB/50021/2020.

Reproducibility

The source code for the experiments in the papers can be downloaded from the Github repository https://github.com/bemali/XD2-Net.

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. URL https://www.tensorflow.org/, Software available from tensorflow.org.

Abdul, A., von der Weth, C., Kankanhalli, M., Lim, B.Y., 2020. COGAM: Measuring and moderating cognitive load in machine learning model explanations. In: Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems. ACM, http://dx.doi.org/10.1145/3313831.3376615.

Alvarez-Melis, D., Jaakkola, T.S., 2018. Towards robust interpretability with self-explaining neural networks. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems. NIPS '18, Curran Associates Inc., Red Hook, NY, USA, pp. 7786–7795.

Bautista, A.D., Wangikar, L., Akbar, S.M.K., 2013. Process mining-driven optimization of a consumer loan approvals process. In: Business Process Management Workshops. Springer Berlin Heidelberg, pp. 219–220. http://dx.doi.org/10.1007/978-3-642-36285-9_24.

Camargo, M., Dumas, M., González-Rojas, O., 2019. Learning accurate LSTM models of business processes. In: Lecture Notes in Computer Science. Springer International Publishing, pp. 286–302. http://dx.doi.org/10.1007/978-3-030-26619-6_19.

Chen, C., Lin, K., Rudin, C., Shaposhnik, Y., Wang, S., Wang, T., 2018. An interpretable model with globally consistent explanations for credit risk. arXiv:1811.12615.

Choi, E., Bahadori, M.T., Kulas, J.A., Schuetz, A., Stewart, W.F., Sun, J., 2016. RETAIN: An interpretable predictive model for healthcare using reverse time attention mechanism. In: Proceedings of the 30th International Conference on Neural Information Processing Systems. NIPS '16, Curran Associates Inc., Red Hook, NY, USA, pp. 3512–3520.

Chollet, F., et al., 2015. Keras. URL https://github.com/fchollet/keras.

Chromik, M., Schuessler, M., 2020. A taxonomy for human subject evaluation of Black-Box Explanations in XAI. In: ExSS-ATEC@IUI.

Demšar, J., Bosnić, Z., 2018. Detecting concept drift in data streams using model explanation. Expert Syst. Appl. 92, 546–559. http://dx.doi.org/10.1016/j.eswa.2017.10.003.

Denisov, V., Belkina, E., Fahland, D., 2018. BPIC'2018: Mining Concept Drift in Performance Spectra of Processes. In: BPI Challenge 2018, http://dx.doi.org/10.4121/uuid:3301445f-95e8-4ff0-981f1f204972, 8th International Business Process Intelligence Challenge, BPIC'18 ; Conference date: 09-09-2018 Through 10-09-2018.

van Dongen, B., 2012. BPI challenge 2012. http://dx.doi.org/10.4121/UUID:3926DB30-F712-4394-AEBC-75976070E91F.

van Dongen, B., 2017. BPI challenge 2017. http://dx.doi.org/10.4121/UUID:5F3067DF-F10B-45DA-B98B-86AE4C7A310B.

van Dongen, B., Borchert, F.F., 2018. BPI challenge 2018. http://dx.doi.org/10.4121/UUID:3301445F-95E8-4FF0-98A4-901F1F204972.

Doshi-Velez, F., Kim, B., 2017. Towards a rigorous science of interpretable machine learning. arXiv, URL https://arxiv.org/abs/1702.08608.

Evermann, J., Rehse, J.-R., Fettke, P., 2017. Predicting process behaviour using deep learning. Decis. Support Syst. 100, 129–140. http://dx.doi.org/10.1016/j.dss.2017.04.003.

Galanti, R., Coma-Puig, B., de Leoni, M., Carmona, J., Navarin, N., 2020. Explainable predictive process monitoring. In: 2nd International Conference on Process Mining, ICPM 2020, Padua, Italy, October 4-9, 2020. IEEE, pp. 1–8.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D., 2019. A survey of methods for explaining black box models. ACM Comput. Surv. 51 (5), 1–42. http://dx.doi.org/10.1145/3236009.

Harl, M., Weinzierl, S., Stierle, M., Matzner, M., 2020. Explainable predictive business process monitoring using gated graph neural networks. J. Decis. Syst. 29 (sup1), 312–327. http://dx.doi.org/10.1080/12460125.2020.1780780.

Hoque, M.N., Mueller, K., 2022. Outcome-explorer: a causality guided interactive visual interface for interpretable algorithmic decision making. IEEE Trans. Vis. Comput. Graphics 28 (12), 4728–4740. http://dx.doi.org/10.1109/TVCG.2021.3102051.

Hsieh, C., Moreira, C., Ouyang, C., 2021. DiCE4EL: Interpreting process predictions using a milestone-aware counterfactual approach. In: 3rd International Conference on Process Mining, ICPM 2021, Eindhoven, the Netherlands, October 31 - Nov. 4, 2021. IEEE, pp. 88–95.

Kraus, M., Feuerriegel, S., 2019. Forecasting remaining useful life: Interpretable deep learning approach via variational Bayesian inferences. Decis. Support Syst. 125, 113100. http://dx.doi.org/10.1016/j.dss.2019.113100.

Krishna, S., Han, T., Gu, A., Pombra, J., Jabbari, S., Wu, S., Lakkaraju, H., 2022. The disagreement problem in explainable machine learning: A practitioner's perspective. CoRR, abs/2202.01602 arXiv:2202.01602 URL https://arxiv.org/abs/2202.01602.

Kwon, B.C., Choi, M.-J., Kim, J.T., Choi, E., Kim, Y.B., Kwon, S., Sun, J., Choo, J., 2019. RetainVis: Visual analytics with interpretable and interactive recurrent neural networks on electronic medical records. IEEE Trans. Vis. Comput. Graph. 25 (1), 299–309. http://dx.doi.org/10.1109/tvcg.2018.2865027.

Le, M., Gabrys, B., Nauck, D., 2014. A hybrid model for business process event and outcome prediction. Expert Syst. 34 (5), e12079. http://dx.doi.org/10.1111/exsy.12079.

Lopes, P., Silva, E., Braga, C., Oliveira, T., Rosado, L., 2022. XAI systems evaluation: A review of human and computer-centred methods. Appl. Sci. 12 (19), 9423. http://dx.doi.org/10.3390/app12199423.

Lundberg, S.M., Lee, S.-I., 2017. A unified approach to interpreting model predictions. In: Proceedings of the 31st International Conference on Neural Information Processing Systems. NIPS '17, Curran Associates Inc., Red Hook, NY, USA, pp. 4768–4777.

Mehdiyev, N., Fettke, P., 2020. Prescriptive process analytics with deep learning and explainable artificial intelligence. In: ECIS.

Mehdiyev, N., Fettke, P., 2021. Explainable artificial intelligence for process mining: A general overview and application of a novel local explanation approach for predictive process monitoring. In: Studies in Computational Intelligence. Springer International Publishing, pp. 1–28. http://dx.doi.org/10.1007/978-3-030-64949-4_1.

Metzger, A., Neubauer, A., Bohn, P., Pohl, K., 2019. Proactive process adaptation using deep learning ensembles. In: Advanced Information Systems Engineering. Springer International Publishing, pp. 547–562. http://dx.doi.org/10.1007/978-3-030-21290-2_34.

Miller, T., 2019. Explanation in artificial intelligence: Insights from the social sciences. Artificial Intelligence 267, 1–38. http://dx.doi.org/10.1016/j.artint.2018.07.007.

Montavon, G., Samek, W., Müller, K.-R., 2018. Methods for interpreting and understanding deep neural networks. Digit. Signal Process. 73, 1–15. http://dx.doi.org/10.1016/j.dsp.2017.10.011.

Narayanan, M., Chen, E., He, J., Kim, B., Gershman, S., Doshi-Velez, F., 2018. How do humans understand explanations from machine learning systems? An evaluation of the human-interpretability of explanation. arXiv:1802.00682.

Nguyen, A., Martínez, M.R., 2020. On quantitative aspects of model interpretability. arXiv:2007.07584.

Nunes, I., Jannach, D., 2017. A systematic review and taxonomy of explanations in decision support and recommender systems. User Model. User-Adapt. Interact. 27 (3–5), 393–444. http://dx.doi.org/10.1007/s11257-017-9195-0.

O'Malley, T., Bursztein, E., Long, J., Chollet, F., Jin, H., Invernizzi, L., et al., 2019. KerasTuner. https://github.com/keras-team/keras-tuner.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E., 2011. Scikit-learn: Machine learning in python. J. Mach. Learn. Res. 12, 2825–2830.

Ribeiro, M.T., Singh, S., Guestrin, C., 2016. "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. KDD '16, Association for Computing Machinery, New York, NY, USA, pp. 1135–1144. http://dx.doi.org/10.1145/2939672.2939778.

Ribera Turró, M., Lapedriza, A., 2019. Can we do better explanations? A proposal of user-centered explainable AI.

Rudin, C., 2019. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. 1 (5), 206–215. http://dx.doi.org/10.1038/s42256-019-0048-x.

Samek, W., Montavon, G., Vedaldi, A., Hansen, L.K., Müller, K.-R. (Eds.), 2019. Explainable AI: Interpreting, Explaining and Visualizing Deep Learning. Springer International Publishing, http://dx.doi.org/10.1007/978-3-030-28954-6.

Sindhgatta, R., Moreira, C., Ouyang, C., Barros, A., 2020a. Exploring interpretable predictive models for business processes. In: Lecture Notes in Computer Science. Springer International Publishing, pp. 257–272. http://dx.doi.org/10.1007/978-3-030-58666-9_15.

Sindhgatta, R., Ouyang, C., Moreira, C., 2020b. Exploring interpretability for predictive process analytics. In: Service-Oriented Computing. Springer International Publishing, pp. 439–447. http://dx.doi.org/10.1007/978-3-030-65310-1_31.

Stevens, A., Smedt, J.D., Peeperkorn, J., 2022. Quantifying explainability in outcome-oriented predictive process monitoring. In: Lecture Notes in Business Information Processing. Springer International Publishing, pp. 194–206. http://dx.doi.org/10.1007/978-3-030-98581-3_15.

Tama, B.A., Comuzzi, M., 2022. Leveraging a heterogeneous ensemble learning for outcome-based predictive monitoring using business process event logs. Electronics 11 (16), 2548. http://dx.doi.org/10.3390/electronics11162548.

Tax, N., Verenich, I., Rosa, M.L., Dumas, M., 2017. Predictive business process monitoring with LSTM neural networks. In: Advanced Information Systems Engineering. Springer International Publishing, pp. 477–492. http://dx.doi.org/10.1007/978-3-319-59536-8_30.

Teinemaa, I., Dumas, M., Rosa, M.L., Maggi, F.M., 2019. Outcome-oriented predictive process monitoring. ACM Trans. Knowl. Discov. Data 13 (2), 1–57. http://dx.doi.org/10.1145/3301300.

Tomsett, R., Braines, D., Harborne, D., Preece, A., Chakraborty, S., 2018. Interpretable to whom? A role-based model for analyzing interpretable machine learning systems. arXiv:1806.07552.

van der Aalst, W.M.P., 2016. Process Mining: Data Science in Action, second ed. Springer.

Velmurugan, M., Ouyang, C., Moreira, C., Sindhgatta, R., 2020. Evaluating explainable methods for predictive process analytics: A functionally-grounded approach. arXiv:2012.04218.

Velmurugan, M., Ouyang, C., Moreira, C., Sindhgatta, R., 2021. Developing a fidelity evaluation approach for interpretable machine learning. arXiv:2106.08492.

Weinzierl, S., Zilker, S., Brunk, J., Revoredo, K., Matzner, M., Becker, J., 2020. XNAP: Making LSTM-based next activity predictions explainable by using LRP. In: Business Process Management Workshops. Springer International Publishing, pp. 129–141. http://dx.doi.org/10.1007/978-3-030-66498-5_10.

Wickramanayake, B., He, Z., Ouyang, C., Moreira, C., Xu, Y., Sindhgatta, R., 2022a. Building interpretable models for business process prediction using shared and specialised attention mechanisms. Knowl.-Based Syst. 248, 108773. http://dx.doi.org/10.1016/j.knosys.2022.108773.

Wickramanayake, B., Ouyang, C., Moreira, C., Xu, Y., 2022b. Generating purpose-driven explanations: The case of process predictive model inspection. In: De Weerdt, J., Polyvyanyy, A. (Eds.), Intelligent Information Systems. Springer International Publishing, Cham, pp. 120–129.

Wolanin, A., Mateo-García, G., Camps-Valls, G., Gómez-Chova, L., Meroni, M., Duveiller, G., Liangzhi, Y., Guanter, L., 2020. Estimating and understanding crop yields with explainable deep learning in the Indian wheat belt. Environ. Res. Lett. 15 (2), 024019. http://dx.doi.org/10.1088/1748-9326/ab68ac.

Zhao, W., Zhao, X., 2014. Process mining from the organizational perspective. In: Wen, Z., Li, T. (Eds.), Foundations of Intelligent Systems. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 701–708.

Zhou, J., Gandomi, A.H., Chen, F., Holzinger, A., 2021. Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics 10 (5), 593. http://dx.doi.org/10.3390/electronics10050593.