Explainable and Effective Process Remaining Time Prediction Using Feature-informed Cascade Prediction Model

Na Guo, Cong Liu, Caihong Li, Qingtian Zeng, Chun Ouyang, Qingzhi Liu, and Xixi Lu

Abstract—*Predictive Process Monitoring* aims to predict the future information of ongoing process executions by leveraging machine and deep learning techniques. One of the tasks is known as *remaining time prediction*, which focuses on predicting the remaining time of ongoing cases. Accurate remaining time prediction can be valuable and important for improving business operations or taking timely interventions to prevent delays. For predicting the remaining time, existing work has used deep learning techniques to achieve high prediction accuracy. However, most of these techniques tend to learn very complex models that are difficult to explain. Systematic feature selection approaches may help improve both the prediction accuracy and the explainability of the model. In this paper, we introduce a feature-informed cascade prediction framework to predict the remaining time. Specifically, we first propose an approach that builds a tree of features by systematically estimating their effects on the remaining time prediction. Next, we use the tree to either automatically select an optimal combination of features or to guide users in this selection process. Each selected feature is correlated with its prediction results in our Feature-informed Cascade Prediction Model (FCPM) for explainability. The proposed approach has been implemented and is made publicly available. Using eight public real-life event logs, the proposed approach is compared to the state-of-the-art approaches in terms of prediction accuracy. In addition, it is demonstrated that our approach visualizes the impact of each input feature in the prediction of individual cases, producing explanations of the prediction results.

Index Terms—Predictive Process Monitoring, Remaining Time Prediction, Feature Selection, Cascade Prediction Model, LSTM.

1 Introduction

With the rapid digital transformation of enterprises, high-quality event data have been stored in various enterprise information systems. *Process mining* [1], [2] is a set of techniques that can extract valuable information from such historical event data, which can help enterprises raise process efficiency and improve their competitiveness of enterprises. Classical process mining techniques aim to uncover knowledge from existing event data by discovering process models, checking conformance against normative models, and improving actual processes [3]. To enable preventive actions and avoid risks promptly, predictive process monitoring techniques have been proposed [4].

Predictive process monitoring focuses on predicting future information on ongoing cases. Existing prediction tasks include the prediction of remaining time, process outcome, next activity, next event duration, suffix, etc. Predicting the remaining time of a running case can help users take early actions and adjust the subsequent execution steps to avoid the risk caused by possible delay. Traditional process-model-based prediction approaches are difficult to ensure high prediction accuracy. Therefore, many approaches have been proposed using machine learning and deep learning [5], [6]. Most of these approaches focus on encoding the activities or prefixes of cases to train the deep models [7]. Compared to classical approaches, these solutions improved the prediction accuracy to a large extent [8]. Some other research exploited the use or engineering of additional features (e.g., resources, accumulated duration, data attributes), but mainly for clustering traces [9]. To the best of our knowledge, little research has focused on investigating the engineering and selection of process features and their effect on the accuracy of remaining time prediction. In addition, deep models are known to be complex and difficult to explain. Therefore, using a large number of features tends to worsen this unexplainable issue. Systematic feature selection may help improve both the accuracy and explainability of the deep model prediction results [10].

In this paper, we propose a feature-informed cascade

This work is supported by the National Key R&D Program of China under Grant 2022ZD0119501, National Natural Science Foundation of China under Grant 52374221, the Taishan Scholars Program of Shandong Province under Grants ts20190936 and Grant tsqn201909109, the Natural Science Excellent Youth Foundation of Shandong Province under Grant ZR2021YQ45, and the Youth Innovation Science and Technology Team Foundation of Shandong Higher School under Grant 2021KJ031. (Corresponding Authors: Cong Liu and Caihong Li)

- *N. Guo is with School of Electrical and Electronic Engineering, Shandong University of Technology, Zibo 255000, China. (e-mail: guona_7@163.com)*
- *C. Liu is with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China, and also with the School of Computer Science and Technology, Shandong University of Technology, Zibo 255000, China. (e-mail: liucongchina@sdust.edu.cn).*
- *C. Li is with the School of Computer Science and Technology, Shandong University of Technology, Zibo 255000, China. (e-mail: lich@sdut.edu.cn)*
- *Q. Zeng is with the College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China. (e-mail: qtzeng@163.com)*
- *C. Ouyang is with the School of Information Systems, Queensland University of Technology, Brisbane, Australia. (e-mail: c.ouyang@qut.edu.au)*
- *Q. Liu is with the Information Technology Group, Wageningen University & Research, The Netherlands. (e-mail: qingzhi.liu@wur.nl)*
- *X. Lu is with the Department of Information and Computing Sciences, Utrecht University, Utrecht, the Netherlands. (email: x.lu@uu.nl)*

prediction framework for remaining time prediction. The main contributions of the paper include:

- a novel feature selection strategy is proposed to support key features extraction in an automatic way; and
- a feature-informed cascade prediction model is introduced to correlate the effects between each input feature and its prediction results for individual cases.

Note that the framework is general and can be instantiated by any existing deep learning models. Without loss of generality, this paper uses LightGBM [11] to instantiate the prediction model in the feature selection strategy to extract effective features as the input of the feature-informed cascade prediction model. Then, the Long Short-Term Memory (LSTM) network [12] is used to implement the feature-informed cascade prediction model as its wide applicability in various application settings.

The rest of this paper is organized as follows. Section 2 presents a review of the related work. Section 3 introduces the background knowledge. Section 4 details the feature-informed cascade prediction framework. Section 5 evaluates the effectiveness and feasibility of the proposed approach using eight real-life event logs. Section 6 demonstrates a case study to analyze and visualize each step of our approach. Section 7 summarizes this paper and presents further research directions.

2 Related Work

This section reviews existing work on predictive process monitoring, feature selection, and explainable prediction.

2.1 Predictive Process Monitoring

Predictive process monitoring focuses on learning prediction models to make accurate predictions for ongoing cases concerning future execution steps, remaining time, outcome, and resource load [13], [14]. Existing predictive process monitoring techniques can be divided into two categories, i.e., process-aware approaches and non-process-aware ones.

Process-aware approaches exploit an explicit representation of a process model to make predictions [15]. van der Aalst *et al.* [16] propose a remaining time prediction method with a marked transition system as the prediction model and develop the support tool FSM analyzer. To improve the expressiveness of discovered models, Rogge-Solti *et al.* [17] use stochastic Petri nets with generally distributed transitions as a business process prediction model to predict the remaining time of process instances. On the one hand, these process-aware prediction approaches may be relatively easier to interpret and understand as an explicit model exists. On the other hand, the prediction results may be less accurate, because the quality of the process model depends on the underlying process discovery algorithm, and the existing discovery techniques may produce low-quality models for processes with complex constructs, e.g., long-term dependency and inclusive choice [18].

Non-process-aware approaches typically use machine learning techniques for learning models and making predictions. Leontjeva *et al.* [19] encode the prefix trace using Hidden Markov Models, and construct an outcome prediction classifier using a random forest (RF) model. By comparing with decision tree, support vector machines (SVM) and generalized boosted regression models, this approach achieves better accuracy. To solve the imbalanced classification problem in the next activity prediction, Kim *et al.* [20] consider five classifiers that often have been adopted, i.e., Artificial Neural Network, Logistic Regression, Gaussian Naive Bayes, RF and SVM to evaluate different resampling approaches, and proposed a novel classification performance measure. This demonstrated the advantages of machine learning technology in a small sample.

With the improvement of data quality and quantity in event logs, deep learning techniques have also been widely used. Tax *et al.* [21] apply LSTM neural network to business process prediction, and add time attributes, i.e., date and week extracted from timestamp information, as input. Compared with process-aware approaches, the deep learning techniques significantly improve the prediction accuracy [8]. For instance, Bukhsh *et al.* [22] apply the Transformer model to the remaining time prediction task, and use activity and time attributes as input, showing the advantages of the advanced model. Ni *et al.* [23] adopt the Encoder-Decoder model to encode activities and obtain effective context relations. Camargo *et al.* [24] used LSTM neural network to build a prediction model, added resource attribute to the input, and adopted the encoding method of word vector. More recently, Pegoraro *et al.* [25] encoded text attributes as one of the inputs, which further improved the accuracy of prediction. Effective attribute encoding helps capture context relationships to improve prediction accuracy. However, existing encoding approaches may mix multiple attributes, which leads to a negative influence on interpretation.

2.2 Feature Selection

Feature selection is a data preprocessing step in machine learning. The objectives of feature selection include building simpler and more comprehensible models, improving data-mining performance, and preparing clean data [26].

Feature selection approaches can be categorized as Filter-based, Embedded, and Wrapper-based [27]. More specifically, filter-based feature selection assigns important values by calculating the correlation between uni-variate or multi-variable and target. In this area, Bommert *et al.* review different filter approaches to compare their performance concerning both run time and predictive accuracy in [28]. Embedded feature selection embeds feature selection approach in the learning model, and the most typical approaches are the boosting algorithm based on the decision tree in [29].Wrapper-based approaches select feature combination by adding/deleting features from the learning model. Forward/backward sequential selection, and recursive feature elimination with cross-validation are discussed as typical Wrapper-based implementations in [30].

Feature selection approaches can be applied to various fields requiring data analysis. Zandkarimi *et al.* [31] propose a generic framework for trace clustering in process mining, taking the feature selection as a key step for trace clustering to avoid problems with overfitting, precision issues, and extra processing costs. Medeiros *et al.* [32] propose process mining based on clustering, adopting the frequency-based

method to select more refined features for partition process instances. Although feature selection has been widely used in the process trace clustering area, less attention has been given to other process mining applications.

2.3 Explainable Prediction

Compared to process-aware approaches, deep-learning-based non-process-aware approaches attract much more attention and are more widely used because of their better prediction accuracy and stability. Most of these approaches are not explainable by design and approaches with better explainability are needed. Sindhgatta *et al.* [10] apply the machine learning explainable technology LIME to the prediction model constructed by XGBoost, and provided a certain explanation for the prediction results. More recently, they propose an interpretable attention-based LSTM model for process behaviour prediction in [33]. Galanti *et al.* use the game theory of Shapley Values to obtain robust explanations for business process prediction in [34]. Hsieh *et al.* [35] design an extended DiCE counterfactual method, which supports the derivation of counterfactual conditions with milestone perception at different stages of the trace to promote explainability. Existing approaches tend to explain the prediction results through post hoc interpretation and visualization from a holistic view. However, the correlation between each input feature and its prediction results is unclear at the case level.

3 Background Knowledge

In this section, we formalize preliminary concepts that are required to describe our approach, such as event logs, process remaining time prediction, and feature encoding.

3.1 Event Logs

An event log is a set of traces where each trace records one execution of the underlying business process. Each trace is composed of chronologically ordered events where each event refers to the execution of an activity. Each activity represents a step in a business process.

Definition 1. (**Event, Attribute**) Let E be the event universe, i.e., the set of all event identifiers. Events may be characterized by various attributes. Let AN be a set of attribute names. For any $e \in E$ and attribute $n \in AN : \#_n(e)$ represents the value of attribute n for event e.

Let U_A be the activity universe, $\#_{act}(e) \in U_A$ is the activity name associated to event e.

Definition 2. (**Trace, Case, and Event log**) A trace is a finite sequence of events, i.e., $\delta \in E^*$, such that each event appears only once and all events are ordered by the timestamp. Let C be the case universe. For any $c \in C$ and attribute name $n \in AN : \#_n(c)$ is the value of attribute n for case c. Each case has a mandatory attribute, i.e., trace, such that $\#_{trace}(c) \in E^*$. An event log L is a set of events such that each event appears at most once in the entire log.

Definition 3. (**Trace prefix**) $tp^k(\delta)$ is a trace prefix of δ, i.e., the sub-sequence consisting of the first k elements of δ.

Table 1 shows a fragment of the public Helpdesk Log[1], where one case with five events is given. Each event has a unique id and a couple of attributes. For example, event e_4 is an instance of activity "Resolve ticket" that occurred on October 25th at 11:54. This event is executed by Peter, and its seriousness is Level 1. Its trace prefix is $\langle e_1, e_2, e_3 \rangle$.

3.2 Business Process Remaining Time Prediction

The remaining time prediction aims to answer business questions like "how long will my business process take to complete?". Predicting the remaining time helps to ensure that the running instance can finish within the required time constraints and enables effective actions to be taken in time without negatively affecting the subsequent process.

The input of the remaining time prediction task is a set of trace prefixes or the last n events, and the output is the required time from the current time to the completion of the instance. Event attributes that are selected as the input of prediction model are called features. If the addition of a feature improves the prediction accuracy, it is defined as a positive feature, and vice versa, a negative feature.

3.3 Feature Encoding

Feature encoding aims to transform categorical feature values to an acceptable data type in model, and the encoding needs to represent the context relationship with high quality.

Label encoding converts each category into an integer value, and it is difficult to effectively distinguish different categories if the number of categories is large when applied to neural network models that cannot support category attributes, e.g., LSTM [21] and Transformer [22]. One-hot encoding applies a binary vector composed of 0 and 1 to represent the feature. The vector dimension refers to the categorical number of the feature, and the position in the vector is set to 1 if the category relates to the feature value. This approach is suitable for a limited number of categories, but it cannot represent the relationship between activities. Word2Vector is a word embedding technology in *natural language processing (NLP)*, including two approaches known as Continuous Bag-of-Words (CBoW) and Skip-gram. CBoW uses the context corpus to train the current word, and Skip-gram uses the current word to train the context. Word2Vector considers the relationship between adjacent activities to represent the relevance between activities.

4 Feature-informed Cascade Prediction Framework

This section first gives an overview of the framework and then explains in detail the feature-informed cascade prediction framework.

4.1 An Overview of the Framework

Fig. 1 shows an overview of the proposed feature-informed cascade prediction framework that consists of the following three steps:

1. https://doi.org/10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb

TABLE 1
A Fragment of the Helpdesk Log

Case id	Event id	Complete Timestamp	Activity	Resource	Seriousness	...
1	e_1	2012/10/9 14:50	Assign seriousness	Peter	Level 1	...
	e_2	2012/10/9 14:51	Take in charge ticket	Peter	Level 1	...
	e_3	2012/10/12 15:02	Take in charge ticket	Mike	Level 1	...
	e_4	2012/10/25 11:54	Resolve ticket	Peter	Level 1	...
	e_5	2012/11/9 12:54	Closed	Niki	Level 1	...
...	...	...	...	...	...	...

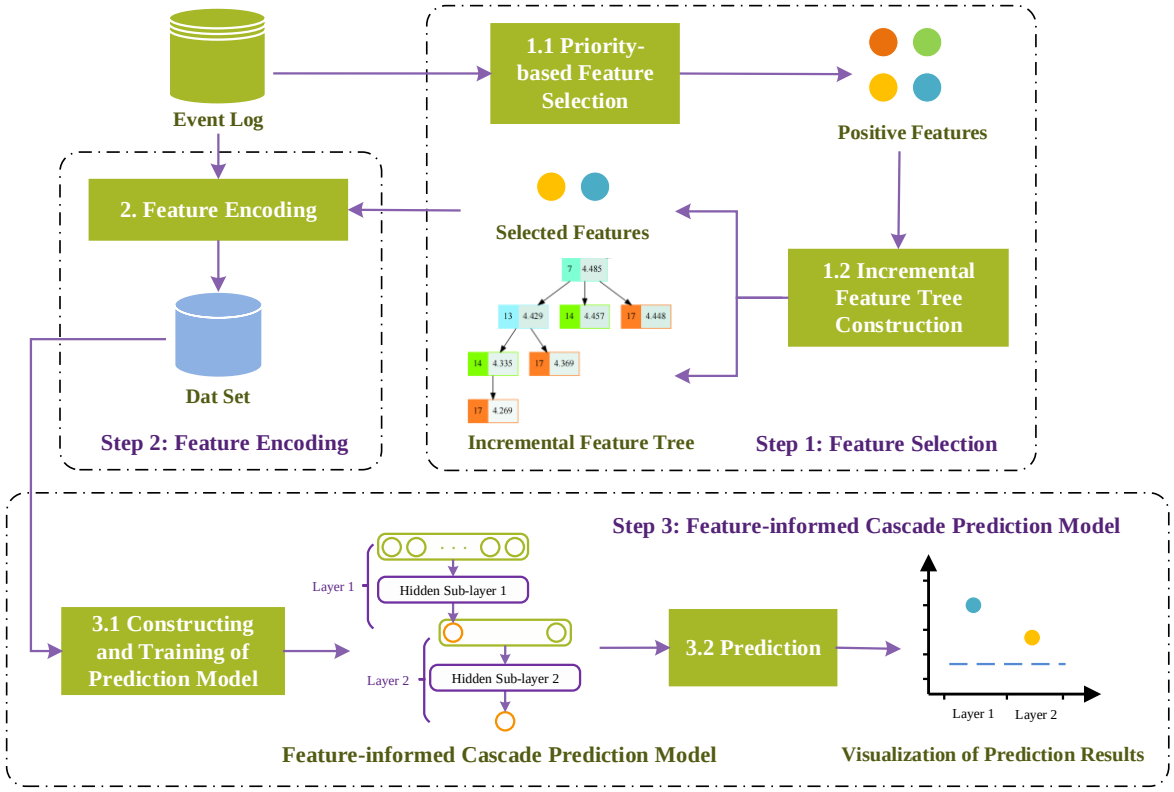


Fig. 1. An Approach Overview.

- **Step 1: Feature Selection (Section 4.2).** By taking an event log as input, an effective feature combination is selected based on the following components:
  - 1.1 **Priority-based Feature Selection.** Features are deleted and added iteratively based on whether they have a positive or negative impact on the prediction results, namely *positive* and *negative features*, respectively. In this step, positive features will be given higher priority and preserved in the selected feature set.
  - 1.2 **Incremental Feature Tree Construction.** Based on the selected feature set, an incremental feature tree is constructed, where each node is a feature. In this feature tree, the path from the root to a node defines a sequence of features with their corresponding prediction results. A sequence of features is called a *feature combination*. Based on the constructed feature tree, the optimal feature combination with acceptable prediction results will be selected.
- **Step 2: Feature Encoding (Section 4.3).** By taking as input an event log and its selected features, word vector encoding is applied for feature encoding.
- **Step 3: Cascade Prediction Model Construction (Section 4.4).** Encoded features are used as the input to construct the prediction model and make predictions as follows:
  - 3.1 **Prediction Model Construction.** Taking an event log and its selected features as input, the feature-informed cascade prediction model is constructed and trained by adding each feature in a sequential manner. Each input feature refers to a dedicated layer including an input sub-layer, a hidden sub-layer, and an output sub-layer. In addition, the input of the first layer has only one feature, and the input of other layers includes a feature and the results of its last layer.
  - 3.2 **Prediction.** Based on the trained model, the output changes of each layer are observed and analyzed to explain the rationality and credibility of the prediction results.

4.2 Feature Selection Strategy

Effective feature selection is important to construct high-quality prediction models. An event log may contain various features and it is challenging to select effective features. As some features may have a high influence on predicting remaining time, it is important to provide a systematic support for selecting them.

In this stage, we examine the importance of each feature by deleting and adding it to a prediction model. The feature selection strategy aims to evaluate the importance of features and the effectiveness of feature combination, and therefore, the proposed strategy has an implicit assumption, i.e., the importance evaluation of features is independent of the underlying prediction model. We use a lightweight model i.e., decision-tree-based boosting algorithm as introduced in [29]. Specifically, the decision tree selects the feature with the largest information gain as the split node, and the information gain indicates the degree of information uncertainty reduction under certain conditions. The information gain of features can be used to quantify the importance of features. In this stage, label encoding is used to encode the categorical features. To evaluate the accuracy of the feature selection strategy, the Mean Absolute Error (MAE) is applied and computed as follows.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y - \hat{y}| \tag{1}$$

As shown in Eq. (1), n represents the number of samples, y represents the true value and $\hat{y}$ is the prediction value. A smaller MAE value indicates a higher prediction accuracy.

Two sequential steps of feature selection strategy are introduced as follows.

First, the priority-based feature selection strategy is introduced. Its main aim is to filter out features that have negative effects on the prediction accuracy. Moreover, we consider the scenario where two or more features may jointly have a positive impact on the prediction results, but each single feature has a negative or a negligible impact. To handle such cases, a backward elimination feature selection strategy is used to avoid damaging the combination scenario among features. If the input feature set contains two or more negative features, the deletion of a negative feature may show a temporary positive impact. Therefore, multiple rounds of feature selection are required. To control the number of rounds of feature selection and ensure that the key feature is not deleted, priority is set for every feature. The activity feature is initialized as the highest priority and priorities of other features are initialized to zero.

Let $F = \{f_1, ..., f_n\}$ be the set of selected features and it is initialized by taking all attributes in the input event log into account. The MAE differences as new feature importance value between two prediction models with and without feature $f_i \in F$ is computed below.

$$I_{MAE}(f_i) = \mathrm{MAE}(F) - \mathrm{MAE}(F - \{f_i\}) \tag{2}$$

As shown in Eq. (2), $\mathrm{MAE}(F)$ represents the MAE value using feature set F, and $\mathrm{MAE}(F-\{f_i\})$ represents the MAE value without feature f_i.

The feature importance set (denoted as I_{dt}) is ranked by the information gain computed using decision tree algorithm. Considering that the information gain of features with multiple results may be large, but they have little contribution to prediction. Therefore, I_{dt} is only treated as the initial reference metric. In each iteration, feature $f_i \in F$ is removed from F if f_i has the lowest priority and the lowest importance value in I_{dt}. Then $I_{MAE}(f_i)$ is computed using Eq. (2). If $I_{MAE}(f_i) < 0$, i.e., f_i is a negative feature, the removal is kept. Otherwise, the removal of f_i is canceled and the priority f_i is boosted. Until $\forall f_i \in F$, $I_{MAE}(f_i) > 0$, the best feature set F is obtained.

Table 2 gives an example to explain the first step such that No. represents the sequence number of the strategy iteration, and MAE represents the obtained results of feature combinations. Given a combination including features 1, 2, 3, 4, 5, 6, and their initial importance is ranked as 1, 3, 6, 4, 5, 2 by I_{dt} in decreasing order. As can be seen from No.1-3, the MAE increases after deleting feature 2, indicating that it is temporarily a positive feature, and therefore, the deletion operation is cancelled. From No.3-4, MAE decreases after deleting feature 5, which indicates it is a negative feature. Based on No.i and No.i+1, the MAE decreases after deleting feature 2 indicating that it is actually a negative feature, but temporarily shows a positive impact due to feature 5. After multiple rounds of iterative operations, features 1, 4, 6, 3 are finally labeled as positive features.

TABLE 2
An Example of the Priority-based Feature Selection Strategy

No.	Feature Combination	MAE
1	1, 2, 3, 4, 5, 6	3.2
2	1, 3, 4, 5, 6	3.3
3	1, 3, 4, 5, 6, 2	3.2
4	1, 3, 4, 6, 2	3.0
...	...	...
i	1, 2, 4, 6, 3	3.0
$i+1$	1, 4, 6, 3	2.9
...	...	...
N	1, 4, 6, 3	2.9

Second, an incremental feature tree is built to visualize the influence of each positive feature on the prediction, which aims to produce the main positive features by excluding features that have a slight positive impact on the prediction results. The algorithm starts from the primary feature Activity and uses it as the root node. The tree expands by adding the next feature f_i with the highest impact $I_{MAE}(f_i)$, iteratively. Each node includes a MAE value that is computed by the training set with features from the root to the current node. After several rounds of iteration, the best feature combination from the root to minimum MAE node in the tree is obtained. On the path of the best feature combination, if the MAE difference between a node and the last node is less than the given threshold, features of the node and its child nodes will be ignored to prevent over-fitting.

Fig. 2 is an incremental feature tree constructed based on the selected feature combination in Table 2. First, the primary feature 1 is used as the root node, and then other features are added according to the strategy. It can be seen that 1, 4, 3, 6 is the best feature combination. If the threshold

is set to 0.2, the three positive features with slight impact, i.e., features 4, 3 and 6, are discarded.

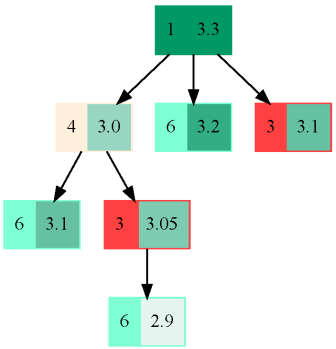


Fig. 2. An Example of the Incremental Feature Tree.

The constructed tree provides additional support, users are capable of balancing between prediction accuracy and efficiency by selecting a dedicated feature set based on the incremental feature tree. As reference, incremental feature tree can provide users with valuable information: (1) feature importance ranking; (2) the contribution of each feature to prediction; (3) the combination relationship between features, as shown in Fig. 2, the prediction error increases by adding feature 6 or 3 to feature combination 1, 4, while adding them together can reduce the error. Therefore, it indicates that features 6 and 3 have a combination relationship.

4.3 Feature Encoding

The encoding approaches mainly come from the word embedding technology in NLP, including Word2Vector, FastText, LSA, Glove, ELMO, GPT and BERT [36], [37]. Word2Vector and FastText are based on local corpus with high optimization efficiency. LSA uses global corpus with high computational complexity. Glove combines the advantages of LSA and Word2Vector. ELMO, GPT and BERT adopt dynamic features to solve the polysemy problem. In business process predictive monitoring application, the number of categorical features is not as large as that in the NLP area, and therefore, Word2Vector is selected from the efficiency and effectiveness point of view.

Predicting the next activity through the prefix of a trace is similar to using context corpus to train the current word, and therefore, the CBoW is used to learn the embedding matrix of the activity feature. Each row of the embedded matrix is a real number vector representing an activity. The activity encoding model is shown in Fig. 3 where the input is the binary vector of the last N activities. E.g., $N = 3$, the last three activities are $[a_1, a_2, a_4]$, the binary vector is $[1, 1, 0, 1, ..., 0, 0]$. The number of the hidden layer units is equal to the vector dimension, and the output is the next activity represented by the One-hot vector. The number of hidden layer units can be tuned for better vector representation. Normally, the higher the prediction accuracy of the CBoW is, the stronger the representation ability of the activity vector is.

For other categorical features whose context maybe not clear, the random vector encoding approach is applied. In this approach, a fixed dimension real number vector

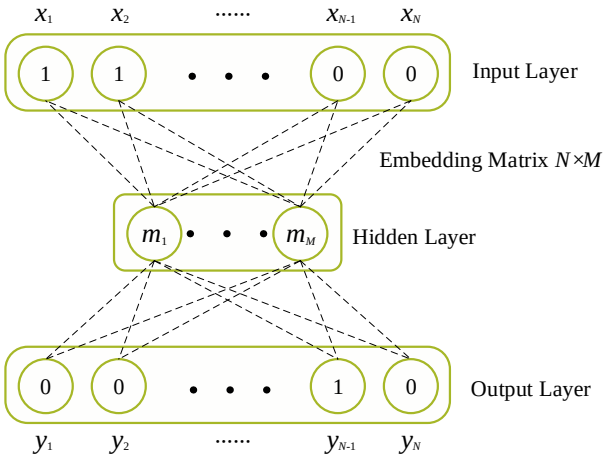


Fig. 3. Activity Encoding Model.

is generated randomly based on the number of feature categories. Note that numerical attributes are handled using a standardized approach.

4.4 Feature-informed Cascade Prediction Model

Deep Neural network (DNN) has been widely used for remaining time prediction. The neural network is composed of the weight, bias, and activation function in neural units. In fact, the influence of each input feature on the prediction results can be explained by the weight of each network cell. However, if the neural network is extremely complex, e.g., millions of parameters are involved, it is difficult to explain how different input features affect the prediction results. To cope with this challenge, a Feature-informed Cascade Prediction Model (FCPM), is proposed by taking as input the selected feature combination.

The architecture of the proposed FCPM is depicted in Fig. 4. For example, the selected features include "Activity", "Duration", and "Resource", and they are ordered descending by their importance. As can be seen in Fig. 4, for layer 1, the model takes prefixes of the feature "Activity" as input, and returns the predicted remaining time as the output of layer 1, after the computation of the hidden sub-layer 1. The embedding layer is designed exclusively for categorical features using the approach in Section 4.3. The hidden layer includes a LSTM layer and a Fully Connection (FC) layer. As for layer 2, the model takes as input the input sub-layer 2 that concatenates the result of the output of layer 1 and the next feature prefixes, which is the "Duration" feature in this example. The prediction result is obtained and given in the output sub-layer 2. For layer 3, its input concatenates the result of the output sub-layer 2 and prefixes of "Resource" feature, and the final prediction result is given in the output sub-layer 3.

Note that the hidden sub-layer in the FCPM can be implemented by any existing network unit. Without loss of generality, in this paper, the hidden sub-layer in the FCPM is implemented based on the LSTM network which is a special type with excellent expression performance of Recurrent Neural Network (RNN) [12], [38]. LSTM network is known as one of the most widely used deep learning solutions for predictive process monitoring because it is

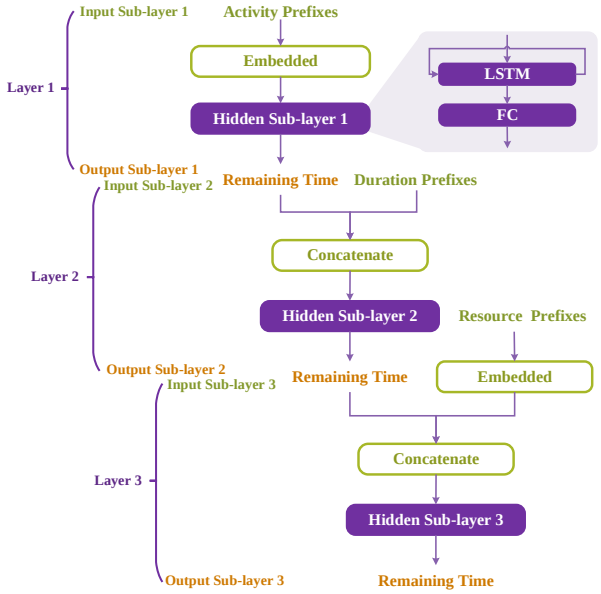


Fig. 4. Architecture of FCPM.

capable of solving the long-term dependency problem of RNN by remembering all prefix information and is good at dealing with time series with long intervals [8].

LSTM has long-term memory depending on its cell structure. Three types of gates, i.e., input gate, forgetting gate, and output gate are included. The value range of the gate output is [0, 1], such that 0 represents the complete discarding of information, and 1 represents the complete retention of information. Eqs. (3)-(8) show the calculated process of LSTM cell. σ represents the sigmoid layer, and $\tanh$ is the hyperbolic tangent layer. x_t, h_t, and C_t represent input, output and cell state in time t, respectively. h_{t-1} and C_{t-1} represent output and cell state in last time $t-1$. Each time sequence input these cells calculated prediction results by the gating information. i_t output by input gate determines how much information in x_t is retained to C_t. f_t output by forgetting gate determines how much information in C_{t-1} is saved to C_t. i_t and f_t as weight calculated C_t by C_{t-1} and the state information $\widetilde{C}_t$. o_t output by output gate determines how much information in C_t is output to h_t. And the weight matrices W_{fh}, W_{ih}, W_{Ch}, W_{oh}, W_{fx}, W_{ix}, W_{Cx}, W_{ox} and bias terms b_f, b_i, b_C, b_o are 12 sets of parameters for training.

$$i_t = \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i) \tag{3}$$

$$f_i = \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f) \tag{4}$$

$$\widetilde{C}_t = \tanh(W_{Ch}h_{t-1} + W_{Cx}x_t + b_C) \tag{5}$$

$$C_t = f_tC_{t-1} + i_t\widetilde{C}_t \tag{6}$$

$$o_t = \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o) \tag{7}$$

$$h_t = O_t\tanh(C_t) \tag{8}$$

As the FCPM has multiple outputs, the back-propagation of all output errors may be difficult to converge during the model training process. Therefore, in the model training process, each batch is trained layer by layer, i.e., layer 1 is trained first, the loss between the predicted value of output sub-layer 1 and the real value is calculated, and the network parameters are adjusted by backpropagation. Then, parameters of the layer 1 are fixed before layer 2 is trained. Next, the loss of output sub-layer 2 is computed. The whole procedure is repeated until all layers are trained.

5 Experimental Evaluation

This section shows the experimental setup and results analysis using eight real-life event logs. Our approach is implemented in Python 3.7. The source code of our implementation and all experimental results are available[2]. The computer configuration used is: Windows11, AMD Ryzen 7 5800H with Radeon Graphics @ 3.20 GHz processor and 16.00GB memory.

5.1 Experimental Settings

Feature selection: In the feature selection stage, seven numerical attributes, i.e., duration, allDuration, year, month, day, week, and hour, are generated using the timestamp. Note that duration refers to the time from the completion of the last event to the completion of the current event. And allDuration refers to the time from the beginning of the case to the completion of the current event. The LightGBM [11], one of the most efficient decision tree based boosting algorithms that can ensure high accuracy, is used to build the prediction model for estimating the impact of features.

Feature encoding: The embedding size of CBoW sets over ranges {4,8,16,32,64} [39]. The dimension of encoding is equal to the embedding size. Based on the experiment results, it is determined that the activity encoding dimension for the BPIC2015 series logs is 32, while for other logs, it is 16. By referring to the CBoW activity encoding dimension, the dimension of random vector encoding is set to encode other categorical features.

Data set partition and trace prefix selection: To ensure that traces of the training set and test set have similar time distribution, the event log is divided into five parts according to the end time sequence of the trace. For each part, one fifth of the traces that complete the earliest are selected as the test set, and the rest of the traces are randomly divided into the training set and the validation set according to the ratio of 4:1. Different prediction models adopt trace prefixes of different lengths. More specifically, the LightGBM algorithm needs to set the last event of the prefix as input in the feature selection stage while the LSTM neural network uses the entire trace prefix as input. In addition, CBoW needs to set a fixed prefix length. According to the accuracy of the next activity prediction, this paper determines that the prefix length of the activity encoding model input is 3 among the prefix length range of {1,3,5}.

Feature-informed Cascade Prediction Model: We use LSTM to implement the model. The hyperparameters of

2. https://github.com/gn874682003/Explainable-Prediction-Framework

the feature-informed cascade prediction model are tuned using the following ranges: (1) the number of hidden layer nodes: {4,8,16,32,64}; (2) batch size: {20,50,100,150}; (3) learning rate: $\{10^{-2}$,10^{-3},$10^{-4}\}$; (4) the number of training epoch: {100,200,300,400,500}; and (5) optimization algorithm: Adam.

5.2 Data set

In our experiments, eight real-life event logs from 4TU Research Center are used as data sets to demonstrate the effectiveness of the proposed approach.

- Helpdesk Log[3] involves the ticket management process of the help desk in an Italian software company.
- BPIC2012 Log[4] is the loan application process of a financial institution.
- BPIC2015 series Logs[5] contains five event logs provided by five Dutch municipal governments. The data includes all building permit applications over a period of four years.
- Production Log[6] comes from the data of product production process in a production workshop.

Table 3 shows a statistical overview of eight event logs, where the number of numerical attributes includes with seven attributes generated using the timestamp. These event logs cover business processes with different data scales and complexity.

5.3 Experimental Results of Feature Selection Strategy

To demonstrate the effectiveness of the proposed feature selection strategy, four different feature selection approaches are set for comparison. The experimental results are shown in Table 4 such that (1) Activity indicates only activity attribute is selected; (2) #All indicates all attributes are selected; (3) FeaNum indicates the number of features; (4) FeaSel indicates features are selected by the proposed feature selection strategy; (5) NullImp indicates features are selected by a corrected feature importance measure as introduced in [40] where the feature importance is not affected by features with more results; and (6) RFECV indicates features are obtained by the recursive feature elimination with cross-validation in [30], which evaluates the feature importance in different data distributions. All the above feature selection strategies are implemented in the LightGBM model. Table 5 shows that all FeaSel features are listed in descending order in terms of importance, and common features are bolded in each log of BPIC2015 series.

The experimental results are described as follows. Compared to Activity, the prediction error of #All is much lower for most of the cases. Exceptions are the BPIC2012 and BPIC2015_4. This can be attributed to the fact that some features have a negative impact on prediction tasks. After filtering the negative and slightly positive features, MAE values of FeaSel are lower than #All excerpt BPIC2015_2, BPIC2015_3 and BPIC2015_5 logs. For BPIC2015 series logs, the number of traces is close to their variant number, i.e., the underlying process varies a lot during execution, indicating that there is a deviation in the data distribution. For these business processes and their corresponding event logs, choosing a reasonable feature selection is extremely challenging. According to Table 4, the results of NullImp and RFECV are not convincing. RFECV aims to select features with high importance while the NullImp focuses on eliminating features that may lead to over-fitting. Both approaches do not fully consider the influence of the feature combination. According to the FeaSel features of the BPIC2015 series logs in Table 5, although the data distribution of these logs varies a lot, the top-ranked features that are bolded in Table 5 are consistent, which indicates that the proposed strategy is reasonable and stable.

Based on the testing results, we can conclude that the proposed strategy is able to find a balance between the prediction accuracy and the number of features for further prediction. In addition, another contribution is to provide users with valuable insights through incremental feature trees. Users can combine feature selection results with domain knowledge to optimize feature combinations.

5.4 Experimental Results of FCPM

In this experiment, we aim to show the effectiveness of selected features, encoding approach and feature-informed cascade prediction model for process remaining time prediction. The experimental settings and results are shown in Table 6 such that (1) LabAct indicates the model takes the activity feature with label encoding as input; (2) OneHotAct indicates the model takes the activity feature with One-hot encoding as input; (3) CBoWAct indicates the model takes the activity feature with CBoW encoding as input; (4) LabFeaSel indicates the model takes a one-dimensional vector concatenated by features including in FeaSel as input, where categorical features are encoded by label encoding; (5) EmbFeaSel indicates the model takes a one-dimensional vector concatenated by features including in FeaSel as input, where the activity feature encoded by CBoW and other categorical features are encoded by random vector encoding; and (6) FCPM is the proposed approach. All the above prediction models are implemented by LSTM.

In general, the prediction results of CBoWAct are better than those of LabAct and OneHotAct, meaning that the CBoW encoding technique can capture the inter-relations among activities. In addition, MAE values of LabFeaSel are smaller than those of LabAct, indicating that the features selected by the proposed strategy can reduce the prediction error. However, for BPIC2015 series logs, MAE values of EmbFeaSel are larger than CBoWAct. One possible explanation is that all inputs are concatenated into a one-dimensional vector, and the prediction model cannot fully capture and utilize the information. The FCPM achieved the best prediction results in all logs, showing that FCPM can effectively use each feature.

By comparing our approach against state-of-the-art approaches, the results and training time are shown in Table 7. Specifically, the following approaches are used: (1)

3. https://doi.org/10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb
4. https://doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f
5. https://doi.org/10.4121/uuid:31a308ef-c844-48da-948c-305d167a0ec1
6. https://doi.org/10.4121/uuid:68726926-5ac5-4fab-b873-ee76ea412399

TABLE 3
Statistics of Event Logs

Data Set	Cases	Events	Activities	Variants	Categorical Attributes	Numerical Attributes	Mean Case Lengths	Mean Case Duration
Helpdesk	4580	21348	14	226	11	7	5	40.859
BPIC2012	13087	262200	36	4366	2	8	20	8.6113
BPIC2015_1	1199	52217	398	1170	16	7	44	95.716
BPIC2015_2	832	44354	410	828	16	7	53	159.981
BPIC2015_3	1409	59681	383	1349	16	7	42	62.266
BPIC2015_4	1053	47293	356	1049	16	7	45	116.871
BPIC2015_5	1156	59083	389	1153	16	7	51	98.362
Production	225	4543	55	221	10	7	20	20.516

TABLE 4
Comparison of Different Feature Selection Srategies

Data sets	Remaining time (MAE: Day)				
	Activity	#All /FeaNum	FeaSel /FeaNum	NullImp /FeaNum	RFECV /FeaNum
Helpdesk	6.827	5.278/18	4.930/4	**4.926**/5	5.323/14
BPIC2012	7.533	8.186/10	**7.356**/3	8.124/9	8.198/2
BPIC2015_1	45.747	39.709/23	**39.517**/16	41.659/12	40.939/6
BPIC2015_2	75.628	**60.952**/23	63.160/17	72.457/17	78.764/2
BPIC2015_3	23.984	**18.425**/23	18.833/17	21.825/13	20.639/4
BPIC2015_4	52.589	53.076/23	**51.153**/19	52.720/16	52.874/7
BPIC2015_5	42.140	**40.411**/23	41.096/15	42.617/12	48.438/2
Production	12.122	8.693/17	**8.517**/9	11.937/4	10.148/6

TABLE 5
Selected Features from Feature Selection Strategy

Data sets	FeaSel Features
Helpdesk	Activity, allDuration, seriousness_2, service_level
BPIC2012	Activity, allDuration, month
BPIC2015_1	**Activity, IDofConceptCase, allDuration, SUMleges, last_phase**,monitoringResource, termName, **parts, year,** Includes_subCases, **resource,** duration, requestComplete, **month,** hour, Responsible_actor
BPIC2015_2	**Activity, SUMleges, allDuration, IDofConceptCase, parts, month,** Responsible_actor, **last_phase,** day, landRegisterID, **year, resource,** Includes_subCases, week, caseProcedure, monitoringResource, hour
BPIC2015_3	**Activity, IDofConceptCase, allDuration, SUMleges, parts, last_phase, year, month,** Includes_subCases, caseProcedure, day, week, termName, **resource,** requestComplete, Responsible_actor, caseStatus
BPIC2015_4	**Activity, IDofConceptCase, SUMleges, allDuration, last_phase, month, year, parts,** day, caseProcedure, **resource,** termName, Responsible_actor, week, landRegisterID, hour, requestComplete, Includes_subCases, duration
BPIC2015_5	**Activity,** landRegisterID, **allDuration, SUMleges, IDofConceptCase, last_phase, parts, month, year,** termName, requestComplete, **resource,** week, hour, day
Production	Activity, Work Order Qty, allDuration, Part Desc., day, month, week, hour, Worker ID

Camargo_LSTM [41], a multi-task prediction model, concatenates all the inputs and completely shares the first LSTM layer to enrich information, which can help to differentiate execution patterns; (2) CRTP_LSTM [42] is a complete remaining trace prediction approach, utilizing all available attributes of previously observed events to predict the complete remaining trace and time; (3) GRU_NP [43] is a remaining time prediction model based on gated RNN, which explains the model by constructing reachability graph; (4) Auto-encoded [23] can improve the prediction accuracy of complex process; and (5) Process Transformer [22] applies Transformer, one of the most advanced deep learning models with excellent fitting and generalization capabilities, and uses activity and time attributes as input, it is suitable for processing large-scale data. We reproduced these approaches using the same experimental setting, e.g., training set and testing set division, as the FCPM.

Generally speaking, FCPM achieves the smallest prediction error, which indicates that the selected features are more effective for high-quality prediction model construction and the cascade input structure can capture all feature information. However, there are some exceptions, i.e., BPIC2015_4 and BPIC2015_5 logs. For BPIC2015_4 and BPIC2015_5 which contains enough prefix states, Auto-encoded can effectively distinguish different prefixes and get the best results. We use the Nemenyi statistical test to evaluate the proposed FCPM and baseline approaches, and the evaluation results are shown in Fig. 5. It can be seen that, FCPM achieves obvious advantages compared to baseline approaches, which demonstrates that an effective feature selection strategy can significantly improve the accuracy even when the prediction is built on basic deep learning models.

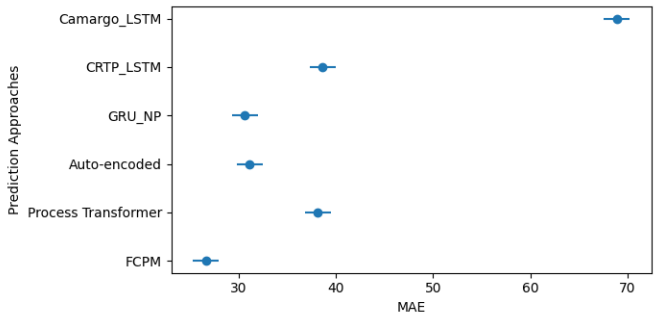


Fig. 5. Approaches Evaluation Using Nemenyi Statistical Test.

From the training time of each prediction model in Table 7, it can be seen that the Auto-encoded approach achieves the best training time while the training time of the Process Transformer is the longest compared to other approaches. One possible reason is that the Process Transformer approach requires the most parameters than others. Among the four approaches based on RNN, although FCPM contains the most abundant input information, its training efficiency is at a medium level.

To observe the earliness [8] of the proposed FCPM, the evaluation results in terms of MAE as the prefix lengths vary

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3353817

TABLE 6
Ablation Experiment of FCPM

Data sets	Remaining time (MAE: Day)					
	LabAct	OneHotAct	CBoWAct	LabFeaSel	EmbFeaSel	FCPM
Helpdesk	6.968	6.626	6.550	5.193	4.858	**4.573**
BPIC2012	7.685	6.466	6.374	7.293	6.341	**6.287**
BPIC2015_1	46.725	38.006	37.723	40.519	40.081	**35.281**
BPIC2015_2	98.076	68.466	65.491	68.856	66.893	**62.017**
BPIC2015_3	20.044	18.424	16.994	17.283	17.404	**16.544**
BPIC2015_4	78.294	51.797	47.995	47.788	55.655	**44.617**
BPIC2015_5	58.905	35.879	36.599	43.163	36.725	**35.604**
Production	11.426	9.648	9.633	10.007	9.556	**8.283**

TABLE 7
Results of FCPM and Other Approaches

Data sets	Camargo_LSTM [41]		CRTP_LSTM [42]		GRU_NP [43]		Auto-encoded [23]		Process Transformer [22]		FCPM	
	MAE	Time(s)	MAE	Time(s)	MAE	Time(s)	MAE	Time(s)	MAE	Time(s)	MAE	Time(s)
Helpdesk	8.546	482s	19.570	1688s	6.544	627s	9.140	99s	5.688	338s	**4.573**	1100s
BPIC2012	7.896	11419s	11.081	90965s	6.369	62057s	7.826	2040s	6.325	13350s	**6.287**	2796s
BPIC2015_1	141.613	1284s	52.546	94230s	41.907	10540s	46.661	586s	59.529	68926s	**35.281**	13227s
BPIC2015_2	89.223	950s	83.138	100497s	71.371	10815s	80.159	487s	93.915	183600s	**62.017**	14532s
BPIC2015_3	28.025	1777s	20.505	133050s	17.291	12901s	19.410	727s	30.462	288182s	**16.544**	14887s
BPIC2015_4	79.969	2029s	62.166	80214s	55.214	9399s	**39.839**	547s	48.327	61209s	44.617	16747s
BPIC2015_5	177.646	2739s	48.795	138313s	37.002	13965s	**35.199**	880s	47.897	108791s	35.604	11401s
Production	18.249	165s	11.127	8330s	9.531	984s	10.857	26s	13.039	963s	**8.283**	437s

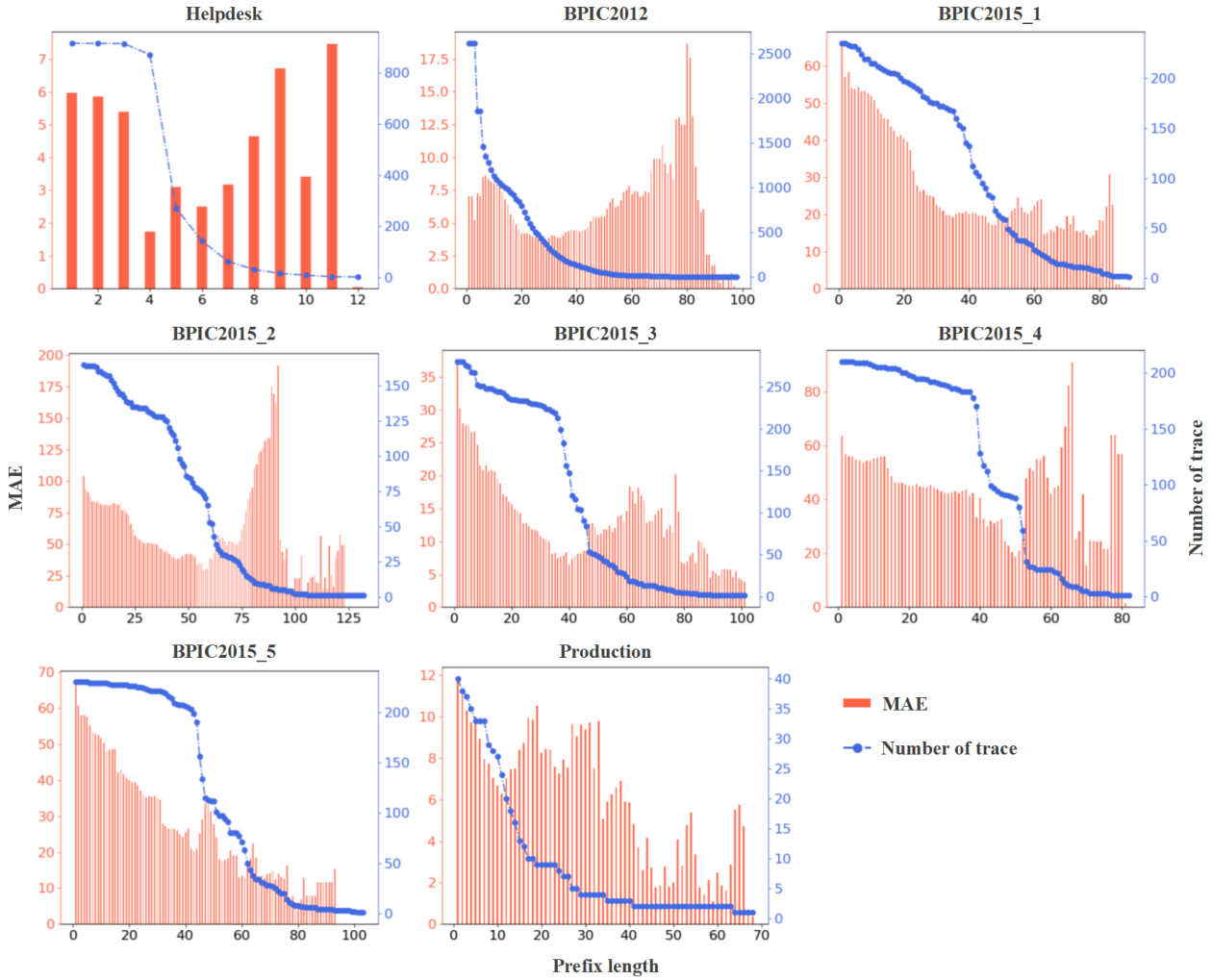


Fig. 6. MAE of Various Prefix Lengths.

are shown in Fig. 6, based on which we see that the MAE of the prediction model gradually decreases and tends to be stable with the increase of prefix length. The number of traces used for evaluation is monotonically decreasing as the prefix length increases. Therefore, as the case progresses, the prediction ability of the FCPM reduces gradually. Generally speaking, the shorter the prefix length when the prediction accuracy reaches the acceptable threshold, the better the prediction model in terms of earliness. By taking the MAE of the FCPM in Table 7 as the acceptable prediction error, the following observations are given. For event logs with relatively short case lengths, e.g., Helpdesk, the prediction result is acceptable when the prefix length is 4, which indicates good earliness. For event logs with relatively long case lengths, e.g., BPIC2012 and BPIC2015 series ones, an acceptable prediction error can be achieved when the prefix length is about 20. Although the Production event log has a small number of cases for model training, it still shows a good prediction trend in the early stage. Based on the above observations, FCPM performs well in terms of earliness in different event logs.

6 Case Analysis

This section takes the Helpdesk Log as a case study to demonstrate the effectiveness of the proposed feature selection strategy and explainability prediction results. Fig. 7 shows the feature importance value obtained based on Step 1.1 (Section 4.2), demonstrating the importance of each feature on the prediction task to provide a global explanation. Because the Activity is the primary feature, and its importance is not calculated. Then, seven features (the Activity and the other six features with positive importance values) are used as input.

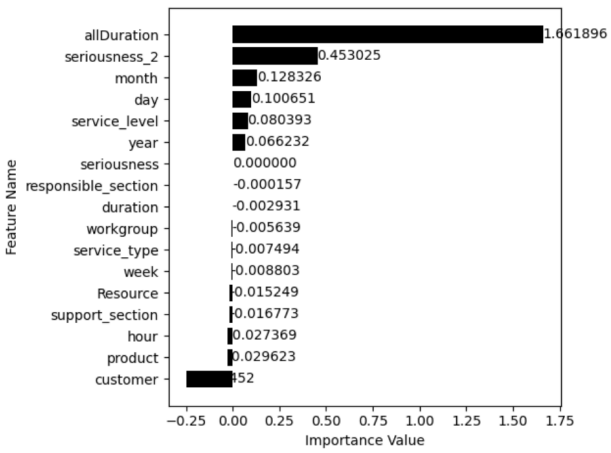


Fig. 7. Feature Importance of Ranking Helpdesk.

Fig. 8 shows the incremental feature tree constructed by taking the seven selected features as input. For each tree node, the number in the left cell represents the index of the feature (e.g., index 0 means Activity feature), and the value in the right cell is the MAE value of the prediction result by using the feature combination from the current node to the root node. The full mapping between indexes and features are: [0: Activity, 6: seriousness_2, 7: service_level, 12: allDuration, 13: month, 14: day, 17: year]. The incremental

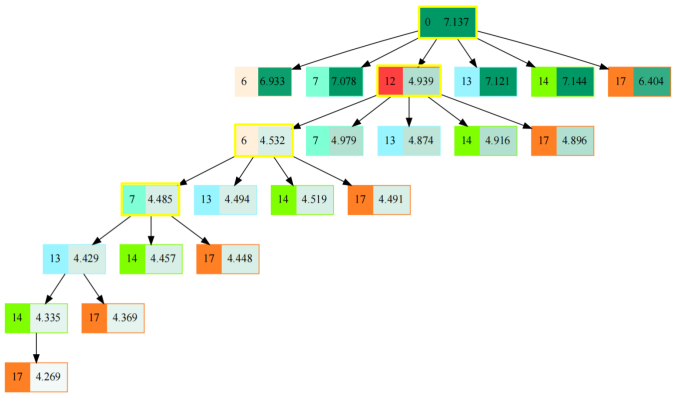


Fig. 8. An Example Incremental Feature Tree with Nine Features.

feature tree is constructed from the root node by selecting the feature of each layer that leads to the smallest MAE value in an iterative way. If a feature is already used, it is not visited in the following layer any more. All the features in the path from the root node to the node with the smallest MAE represent the best feature combination. Because there may be distribution deviation between the verification set and the test set, the important features are [0: Activity, 12: allDuration, 6: seriousness_2, 7: service_level] by deleting the features with slight impact according to the threshold (set to 0.2). Then, the four features are used as input, and the obtained MAE=4.930 in the test set. The incremental feature tree helps visualize the obtained MAE of all feature combinations, which can provide users with selection assistance. In addition, this tree helps to understand the effect of combination relationships among features. For example, considering the third layer, adding feature 7: service_level can increase the prediction error, but adding the feature after feature 6: seriousness_2 can reduce the prediction error. It can be concluded that features 6 and 7 may have an implicit inter-dependency between each other.

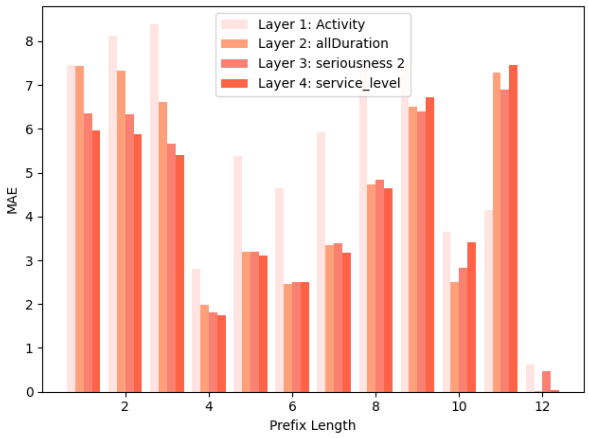
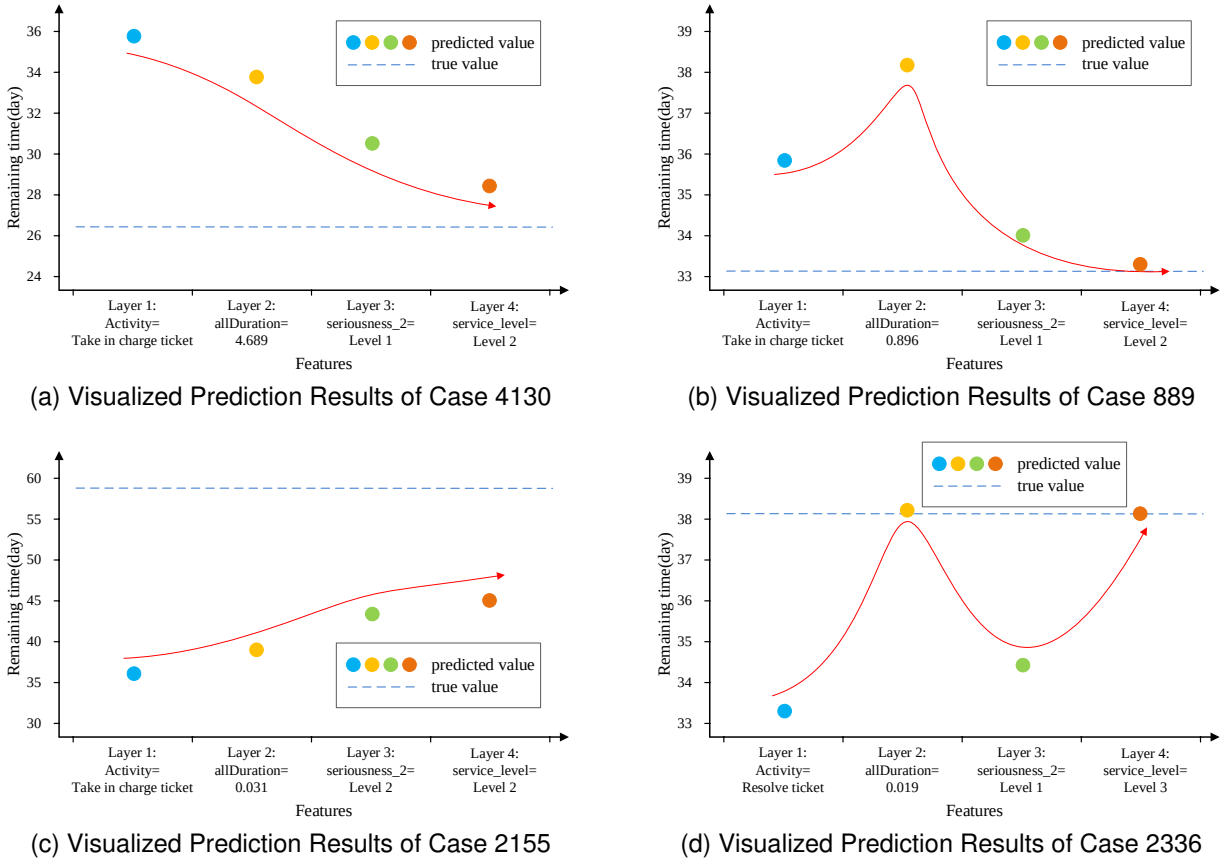


Fig. 9. MAE of Each Input Layer in Various Prefix Lengths.

We selected the top four features for the Helpdesk event log according to Fig. 8 for the FCPM training. The overall prediction error decreased layer by layer and MAE values are 6.559, 5.537, 4.842, and 4.573, respectively. For different trace prefix lengths, the contribution of the four features to

This article has been accepted for publication in IEEE Transactions on Services Computing. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TSC.2024.3353817



(a) Visualized Prediction Results of Case 4130

(b) Visualized Prediction Results of Case 889

(c) Visualized Prediction Results of Case 2155

(d) Visualized Prediction Results of Case 2336

Fig. 10. Examples of FCPM Prediction Results, (a) Cid='Case 4130', trace prefix=⟨('Assign seriousness', 0.0, 'Level 1', 'Level 2'),('Take in charge ticket', 4.689, 'Level 1', 'Level 2')⟩; (b) Cid='Case 889', trace prefix=⟨('Assign seriousness', 0.0, 'Level 1', 'Level 2'),('Take in charge ticket', 0.896, 'Level 1', 'Level 2')⟩; (c) Cid='Case 2155', trace prefix=⟨('Assign seriousness', 0.0, 'Level 2', 'Level 2'),('Take in charge ticket', 0.031, 'Level 2', 'Level 2')⟩; (d) Cid='Case 2336', trace prefix=⟨('Assign seriousness', 0.0, 'Level 1', 'Level 3'),('Resolve ticket', 0.019, 'Level 1', 'Level 3')⟩.

the remaining time prediction is shown in Fig. 9. Based on this, we can see that in the early stage of the process, each feature has a positive impact on the prediction accuracy, and as the process progresses, the positive influence of seriousness_2 and service_level becomes weak. Due to the number of traces in the event log is monotonically decreasing as the prefix length increases, the prediction ability of the model may decrease gradually, which is a common defect of deep learning models. Generally speaking, based on the MAE comparison, it is demonstrated that effective features input can significantly improve prediction accuracy even when the prediction is built on very basic deep learning models.

To illustrate the explainability of the proposed feature-informed cascade prediction model, four cases with IDs 4130, 889, 2155 and 2336 in the Helpdesk Log are visualized and analyzed. The prefix of these four cases from the activity point of view are identical, and the prediction is made after the second event with activity name "Take in charge ticket". Fig. 10 shows the prediction results of the four cases using the top-4 features (i.e., Activity, allDuration, seriousness_2 and service_level) according to Fig. 8. The predicted remaining time by adding each input feature incrementally to different layers are visualized. Based on Fig. 10 (a) and (b), we found: (1) For layer 1, the Activity feature is used and both feature values are "Take in charge ticket", and therefore, the predicted values are also identical; (2) For layer 2, the allDuration feature is included and its values are different (4.689 days vs 0.896 day). The predicted remaining time is different for these two cases, i.e., one is closer to the true value, while the other is not. One possible explanation could be that with the increase of the allDuration feature value, the remaining time of the case decreases, and vice versa. Based on Fig. 10 (a) and (c), for layer 3, the seriousness_2 feature is included in the FCPM with different values (Level 1 vs Level 2), the predicted results are closer to the true value compared to the results obtained based on the first two layers. This is consistent with our domain knowledge, i.e., the higher the seriousness level, the longer the time required. Based on Fig. 10 (b) and (d), for layer 4, both feature values and prediction results of the first three layers are similar, but there is a significant difference between the two true values. However, the service_level feature is added and its values are different (Level 2 vs Level 3), both predicted results approximate the true value. The rationale is that the higher the service level, the longer the time required.

Using the FCPM to predict the remaining time of an online instance, the prediction results of each layer can be obtained at the currently event. If the prediction trend of each layer conforms to business regulations and conventional perception, the prediction results can be considered credible and used as a basis for decision-making. Otherwise,

it indicates that the instance may have special circumstances and refuse to trust the prediction result.

7 Conclusion

Deep learning-based business process remaining time prediction has demonstrated its strong capability of achieving high prediction accuracy. Most existing remaining time prediction approaches tend to rely on the activity attribute or the prefix information, which are not sufficient in general cases. In this paper, we proposed a feature selection approach. In addition, a feature-informed cascade prediction model is designed to correlate the effects between each input feature and its prediction results. We have implemented our approach and evaluated its performance using eight real-life event logs. The experimental results have shown that our proposed framework can achieve better prediction accuracy. In addition, based on our model, the relative impact of each attribute is observed and analyzed to explain the rationality and credibility of the prediction results.

The proposed approach can be further strengthened from the following aspects: (1) in terms of control flow, its influence on the remaining time prediction should be explained; and (2) the cascade prediction model needs to accelerate convergence if too many features are selected. In addition, our future work mainly lies in generalizing the explainable prediction framework with more advanced deep learning models to further improve the prediction accuracy and universality. In addition, due to the complexity and diversity of real-life event logs, we prefer to come up with a dedicated process-oriented feature selection approach. Besides explainability from the feature perspective, we would like to continuously explore other advanced explainable models to further improve the prediction reliability.

References

[1] W. M. van der Aalst, *Process mining : discovery, conformance and enhancement of business processes*. Germany: Springer, 2011.

[2] C. Liu, H. Duan, Q. Zeng, M. Zhou, F. Lu, and J. Cheng, "Towards comprehensive support for privacy preservation cross-organization business process mining," *IEEE Transactions on Services Computing*, vol. 12, no. 4, pp. 639–653, 2016.

[3] L. Wen, J. Wang, W. M. van der Aalst, B. Huang, and J. Sun, "A novel approach for process mining based on event types," *Journal of Intelligent Information Systems*, vol. 32, no. 2, pp. 163–190, 2009.

[4] J. C. Buijs, B. F. van Dongen, and W. M. van der Aalst, "A genetic algorithm for discovering process trees," in *2012 IEEE Congress on Evolutionary Computation*. IEEE, 2012, pp. 1–8.

[5] N. Harane and S. Rathi, "Comprehensive survey on deep learning approaches in predictive business process monitoring," *Modern Approaches in Machine Learning and Cognitive Science: A Walkthrough*, pp. 115–128, 2020.

[6] D. A. Neu, J. Lahann, and P. Fettke, "A systematic literature review on state-of-the-art deep learning methods for process prediction," *Artificial Intelligence Review*, pp. 1–27, 2021.

[7] E. Rama-Maneiro, J. C. Vidal, and M. Lama, "Deep learning for predictive business process monitoring: Review and benchmark," *IEEE Transactions on Services Computing*, vol. 16, no. 1, pp. 739–756, 2023.

[8] I. Verenich, M. Dumas, M. L. Rosa, F. M. Maggi, and I. Teinemaa, "Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 4, pp. 1–34, 2019.

[9] J. Evermann, T. Thaler, and P. Fettke, "Clustering traces using sequence alignment," in *International Conference on Business Process Management*. Springer, 2016, pp. 179–190.

[10] R. Sindhgatta, C. Ouyang, C. Moreira, and Y. Liao, "Interpreting predictive process monitoring benchmarks," *arXiv preprint arXiv:1912.10558*, 2019.

[11] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "Lightgbm: A highly efficient gradient boosting decision tree," *Advances in neural information processing systems*, vol. 30, 2017.

[12] A. Graves, "Long short-term memory," *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

[13] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, "Predictive monitoring of business processes: a survey," *IEEE Transactions on Services Computing*, vol. 11, no. 6, pp. 962–977, 2017.

[14] I. Teinemaa, M. Dumas, M. L. Rosa, and F. M. Maggi, "Outcome-oriented predictive process monitoring: review and benchmark," *ACM Transactions on Knowledge Discovery from Data (TKDD)*, vol. 13, no. 2, pp. 1–57, 2019.

[15] A. Bolt and M. Sepúlveda, "Process remaining time prediction using query catalogs," in *International Conference on Business Process Management*. Springer, 2013, pp. 54–65.

[16] W. M. van der Aalst, M. H. Schonenberg, and M. Song, "Time prediction based on process mining," *Information systems*, vol. 36, no. 2, pp. 450–475, 2011.

[17] A. Rogge-Solti and M. Weske, "Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays," in *International conference on service-oriented computing*. Springer, 2013, pp. 389–403.

[18] A. Augusto, R. Conforti, M. Dumas, M. L. Rosa, F. M. Maggi, A. Marrella, M. Mecella, and A. Soo, "Automated discovery of process models from event logs: Review and benchmark," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 4, pp. 686–705, 2019.

[19] A. Leontjeva, R. Conforti, C. Di Francescomarino, M. Dumas, and F. M. Maggi, "Complex symbolic sequence encodings for predictive monitoring of business processes," in *Business Process Management*, H. R. Motahari-Nezhad, J. Recker, and M. Weidlich, Eds. Cham: Springer International Publishing, 2015, pp. 297–313.

[20] J. Kim and M. Comuzzi, "A diagnostic framework for imbalanced classification in business process predictive monitoring," *Expert Systems with Applications*, vol. 184, p. 115536, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741742100943X

[21] N. Tax, I. Verenich, M. L. Rosa, and M. Dumas, "Predictive business process monitoring with lstm neural networks," in *International Conference on Advanced Information Systems Engineering*. Springer, 2017, pp. 477–492.

[22] Z. A. Bukhsh, A. Saeed, and R. M. Dijkman, "Processtransformer: Predictive business process monitoring with transformer network," *arXiv preprint arXiv:2104.00721*, 2021.

[23] W. Ni, M. Yan, T. Liu, and Q. Zeng, "Predicting remaining execution time of business process instances via auto-encoded transition system," *Intelligent Data Analysis*, vol. 26, no. 2, pp. 543–562, 2022.

[24] M. Camargo, M. Dumas, and O. González-Rojas, "Learning accurate lstm models of business processes," in *International Conference on Business Process Management*. Springer, 2019, pp. 286–302.

[25] M. Pegoraro, M. S. Uysal, D. B. Georgi, and W. M. van der Aalst, "Text-aware predictive monitoring of business processes," *arXiv preprint arXiv:2104.09962*, 2021.

[26] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Computers & Electrical Engineering*, vol. 40, no. 1, pp. 16–28, 2014.

[27] R. Zebari, A. Abdulazeez, D. Zeebaree, D. Zebari, and J. Saeed, "A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction," *J. Appl. Sci. Technol. Trends*, vol. 1, no. 2, pp. 56–70, 2020.

[28] A. Bommert, X. Sun, B. Bischl, J. Rahnenführer, and M. Lang, "Benchmark for filter methods for feature selection in high-dimensional classification data," *Computational Statistics & Data Analysis*, vol. 143, p. 106839, 2020.

[29] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[30] P. Misra and A. S. Yadav, "Improving the classification accuracy using recursive feature elimination with cross-validation," *Int. J. Emerg. Technol*, vol. 11, no. 3, pp. 659–665, 2020.

[31] F. Zandkarimi, J.-R. Rehse, P. Soudmand, and H. Hoehle, "A generic framework for trace clustering in process mining," in *2020 2nd International Conference on Process Mining (ICPM)*. IEEE, 2020, pp. 177–184.

[32] A. K. A. d. Medeiros, A. Guzzo, G. Greco, W. M. van der Aalst, A. Weijters, B. F. v. Dongen, and D. Sacca, "Process mining based on clustering: A quest for precision," in *International conference on business process management*. Springer, 2007, pp. 17–29.
[33] R. Sindhgatta, C. Moreira, C. Ouyang, and A. Barros, "Exploring interpretable predictive models for business processes," in *Business Process Management: 18th International Conference, BPM 2020, Seville, Spain, September 13–18, 2020, Proceedings 18*. Springer, 2020, pp. 257–272.
[34] R. Galanti, B. Coma-Puig, M. de Leoni, J. Carmona, and N. Navarin, "Explainable predictive process monitoring," in *2020 2nd International Conference on Process Mining (ICPM)*. IEEE, 2020, pp. 1–8.
[35] C. Hsieh, C. Moreira, and C. Ouyang, "Dice4el: Interpreting process predictions using a milestone-aware counterfactual approach," in *2021 3rd International Conference on Process Mining (ICPM)*, 2021, pp. 88–95.
[36] Y. Goldberg and O. Levy, "word2vec explained: deriving mikolov et al.'s negative-sampling word-embedding method," *arXiv preprint arXiv:1402.3722*, 2014.
[37] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou, and T. Mikolov, "Fasttext. zip: Compressing text classification models," *arXiv preprint arXiv:1612.03651*, 2016.
[38] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.
[39] J. Evermann, J.-R. Rehse, and P. Fettke, "Predicting process behaviour using deep learning," *Decision Support Systems*, vol. 100, pp. 129–140, 2017, smart Business Process Management. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167923617300635
[40] A. Altmann, L. Toloşi, O. Sander, and T. Lengauer, "Permutation importance: a corrected feature importance measure," *Bioinformatics*, vol. 26, no. 10, pp. 1340–1347, 2010.
[41] M. Camargo, M. Dumas, and O. González-Rojas, "Learning accurate lstm models of business processes," in *Business Process Management*, T. Hildebrandt, B. F. van Dongen, M. Röglinger, and J. Mendling, Eds. Cham: Springer International Publishing, 2019, pp. 286–302.
[42] B. R. Gunnarsson, S. v. Broucke, and J. De Weerdt, "A direct data aware lstm neural network architecture for complete remaining trace and runtime prediction," *IEEE Transactions on Services Computing*, vol. 16, no. 4, pp. 2330–2342, 2023.
[43] C. Rui, "Explainable business process remaining time prediction using reachability graph," p. 625, 2023. [Online]. Available: https://cje.ejournal.org.cn/en/article/doi/10.23919/cje.2021.00.170

Na Guo received her MS degree from Shandong University of Technology, Zibo, China in 2021. She is current working towards her PhD degree in the School of Electrical and Electronic Engineering in Shandong University of Technology. Her research interests are in the areas of process mining and process predictive monitoring.

Cong Liu received the BS and the MS degree in computer software and theory from Shandong University of Science and Technology, Qingdao, China. He received the PhD degree in the Department of Mathematics and Computer Science, Eindhoven University of Technology, in 2019. He has been a Full Professor with the Shandong University of Technology since 2019. His research interests are in the areas of process mining, and Petri nets.

Caihong Li received the MS degree in computer software and theory from Shandong University of Science and Technology, Qingdao, China, in 2000. She received the PhD degree in the Shandong University, Jinan, China, in 2007. She is currently a Professor with Shandong University of Technology, Zibo, China. Her research interests are in the areas of intelligent robot and computational intelligence learning algorithm.

Qingtian Zeng received the BS degree and the MS degree in computer science from Shandong University of Science and Technology, Taian, China, and the PhD degree in computer software and theory from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. He is currently a Professor with Shandong University of Science and Technology, Qingdao, China. His research interests are in the areas of Petri nets, process mining, and knowledge management.

Ouyang Chun received the PhD degree in Computer Systems Engineering from University of South Australia. She is currently a associate professor with Queensland University of Technology, Brisbane, Australia. Her research interests are in the areas of data-driven process analytics, process automation, and recently explainable predictive analytics.

Qingzhi Liu received a B.E. and a M.Eng. from Xidian University, China. He received a MS (with cum laude) and a PhD from Delft University of Technology, The Netherlands. He is a Lecturer at the Information Technology Group, Wageningen University & Research, The Netherlands. His research interests include Internet of Things and machine learning.

Xixi Lu received the MS and PhD degree in the Department of Mathematics and Computer Science, Eindhoven University of Technology. She is a Assistant Professor in Utrecht University, in 2019. Her research interests are in the areas of process mining and data science.