Journal of Autonomous Intelligence (2023) Volume 6 Issue 1
doi: 10.32629/jai.v6i1.639

ORIGINAL RESEARCH ARTICLE

Prediction method of business process remaining time based on attention bidirectional recurrent neural network

Ali Fakhri Mahdi Al-Jumaily[1,2*], Abdulmajeed Al-Jumaily[3], Saba Jasim Al-Jumaili[4]

[1] *Education Department - Accounts-Fallujah, General Directorate of Anbar Education, Ministry of Education, Heet, Anbar 31007, Iraq. E-mail: allawy850@gmail.com*

[2] *Faculty of Entrepreneurship and Business, Universiti Malaysia Kelantan, Kota Bharu 16100, Kelantan, Malaysia.*

[3] *Department of Signal Theory and Communications, Universidad Carlos III de Madrid, Madrid 28903, Spain.*

[4] *Laboratory of Climate–Smart Food Crop Production, Institute of Tropical Agriculture and Food Security, Universiti Putra Malaysia (UPM), Selangor 43400, Malaysia.*

ABSTRACT

Most of the existing deep learning-based business process remaining time prediction methods use traditional long-short-term memory recurrent neural networks to build prediction models. Due to the limited modeling ability of traditional long-short-term memory recurrent neural networks for sequence data, and existing methods there is still much room for improvement in the prediction effect. Aiming at the shortcomings of existing methods, this paper proposes a business process remaining time prediction method based on attention bidirectional recurrent neural network. The method uses a bidirectional recurrent neural network to model the process instance data and introduces an attention mechanism to automatically learn the weights of different events in the process instance. In addition, in order to further improve the learning effect, an iterative learning strategy is designed based on the idea of transfer learning, which builds remaining time prediction models for process instances of different lengths, which improves the pertinence of the model. The experimental results show that the proposed method has obvious advantages compared with traditional methods.

Keywords: Prediction Method; Business Process; Remaining Time Prediction; Attention Bidirectional Recurrent Neural Network

ARTICLE INFO

Received: 17 May, 2023
Accepted: 31 May, 2023
Available online: 30 June, 2023

1. Introduction

In recent years, the subject of business process management has seen an attracted interest in the hot new research area of predictive process monitoring. Execution status, such as the next activity to be executed, the final execution result of the instance, the remaining execution time of the instance, etc. Compared with traditional process monitoring methods based on dashboards and reports, predictive process monitoring can not only monitor the execution status of business process instances in real time, but also intelligently predict their possible future execution results, so as to provide necessary manual intervention and provide a basis for decision-making, which helps to avoid adverse consequences such as execution timeout and resource conflict that may be caused by the continued execution of the business process instance. Predicting the remaining process time of a running business process instance is the objective of the significant class of predictive process monitoring activities known as business process remaining time prediction[1,2].

Remaining time prediction plays an important role in the optimi-

zation of business system performance. Taking the hospital visit process as an example, if the patient's visit time can be accurately predicted, the patient's visit experience can be effectively improved, and it can also provide decision-making basis for hospital diagnosis and treatment process optimization. The traditional remaining time prediction research mainly mines formal process models such as transition systems[1,3], stochastic Petri nets[4], and process trees[5] from historical business process execution logs, and then calculates the remaining time according to the process model prediction. In recent years, many researchers have applied machine learning techniques to the task of remaining time prediction, and achieved good prediction results. Verenich *et al.*[6] proposed a remaining time prediction method based on Long-Short Term Memory (LSTM) recurrent neural network, which beat the existing methods based on process model and machine learning and achieved the best prediction effect. Tax *et al.*[7] show that deep learning techniques have broad application prospects in the task of remaining time prediction.

However, the research on remaining time prediction based on deep learning has just started, and there are still many problems that have not been well solved. In order to improve the impact of significant events on the prediction results and remove the interference of unimportant events, the network model takes into account the bidirectional information of the sequence data and applies the attention mechanism. So that modeling of the process instances can be done more effectively. In addition, this paper adopts the transfer learning method to build remaining time prediction models for process instances of different lengths to improve the pertinence of the model. In order to overcome the difference in the number of process instances of different lengths and effectively utilize the correlation between instance prefixes of different lengths, the model is first trained on a large number of short process instances, and then the short-instance model parameters are used as the initial values of the long-instance model, to achieve efficient training of long-instance models by means of Fine-tuning. Finally, this paper conducts experiments on five public event log datasets, and the experimental results show that our method significantly outperforms traditional model-driven and data-driven methods.

2. Related work

The related research on remaining time prediction can be categorized into three groups based on whether it relies on the business process model: model-driven method, data-driven method, and technique integrating model and data. The model-driven method relies on mining the log or predicting the remaining time from the business process model formulated by domain experts in advance. Teinemaa *et al.*[1] proposed a method for predicting the remaining time based on the transition system. The method first mines the transition system model from the event log to record all possible states of the process instance, and then marks the time information on each state in the transition system model, and then predicts its possible state according to the current state of the executing process instance. Remaining execution time. Senderovich *et al.*[4] mined random Petri nets from event logs, simulated the process instance being executed on it, and predicted the remaining execution time according to the simulation results. Jimenez-Ramirez *et al.*[8] used a declarative model to solve the problem.

The remaining time prediction problem is in complex scenarios such as multiple instances and resource constraints. Senderovich *et al.*[9] used the queuing model to solve the remaining time prediction problem when multiple service processes are in a queuing state. The data-driven approach applies machine learning techniques to mine remaining time prediction models directly from historical event logs. The basic approach employed in this type of work is to first use clustering techniques to divide historical process instances, each division representing a class of variants of the process, and then apply regression techniques to build remaining time prediction models on each division[6]. For machine learning methods such as clustering and regression, designing effective instance features is an important factor affecting the model effect. Folino *et al.*[10] conducted research on the feature design problem in the remaining time prediction task.

Since the process instance is a type of sequence data and there are many factors that affect how much time is left, it is necessary to design a sequence encoding scheme to construct the feature vector of the process instance, which must typically be carried out manually by relying on expert experience[11]. In recent years, researchers have used the LSTM network to build a remaining time prediction model, which has achieved good prediction results on real data[7,12]. One advantage of deep learning-based remaining time prediction methods is the ability to automatically learn feature representations for process instances, avoiding it eliminates the large amount of tedious tasks brought by the feature engineering required by traditional machine learning methods.

The major goal of the method of model and data combination is to strengthen the business process model and increase the precision of time forecasts by using machine learning technology. Such techniques are a prominent topic in current research on remaining time forecasting because they can address some of the drawbacks of techniques that merely rely on models or data. Polato *et al.*[13] proposed a data-aware transition system, constructed a naive Bayesian classification model for each state, and constructed a support vector regression model for each transition to predict the remaining time; Verenich *et al.*[14] Using the process tree as an abstract form for business processes, a regression model and a classification model are trained on the activity node and gateway node of the process tree for remaining time prediction, respectively.

3. Proposed model

3.1 Description of business process

The description of the task of predicting the remaining time of a business process is based on the aforementioned basic concepts and notation. Under the machine learning framework, the goal of the business remaining time prediction task is to use the historical process instance data recorded in the event log to train the remaining time prediction model f: $\varepsilon^* \rightarrow 0^+$, which can predict the remaining execution time for the executing process instance (i.e., the trajectory prefix). Further, the remaining time prediction model training can be decomposed into two stages: training set generation and model learning.

The goal of the training set generation phase is to convert event logs into training data that can be used by machine learning algorithms.

The goal of the model learning phase is to learn the remaining time prediction model based on the generated training set. This phase can be abstracted as a regression prediction of the problem, as Equation (1).

$$f^* = \frac{argmin}{f \in F} \sum_{(\sigma,t)\in D} (f(\sigma) - t)^2 + \Omega(f) \tag{1}$$

where, D presents the training samples in the training dataset, σ the training dataset of all lengths is finally returned. The basic idea of the algorithm is to intercept each historical track σ in the event log within the specified length range $[1_{min}, 1_{max}]$, and obtain various track prefixes σ_1 with different lengths and their corresponding remaining time remain $(\sigma, 1)$.

The first part of this optimization problem is the prediction error on the training set, calculated using the mean squared error; the second term is the model regularization term, which is used to overcome the overfitting problem in the learning process.

The fundamental ideas associated with the remaining time forecast task are listed below, and they mostly include business process management concepts such as events, tracks, track prefixes, process instances, and event logs.

The definitions are as follows:

Define 1 event: An event is an execution instance of an activity in a business system, which can be represented as a tuple $e(a, cid, start, end, p_1, \ldots, p_m)$. Where, $a \in A$ is the activity executed in the event, $cid \in N^+$ is the ID of the process instance to which the event belongs, and $end \in N^+$ are the start time and end time of the event execution, respectively ($start < end$), $p_1 \in P_1, \ldots, \in P_m$, which are attributes of the event, such as the executor of the event, execution cost, etc. The event space can be denoted as $\varepsilon = A \times N^+ \times N^+ \times N^+ \times P_1 \ldots \times P_m$.

Define 2 trajectories: A trajectory is a finite non-empty sequence $\sigma = (e_1, \ldots, e_{|\sigma|})$ of events, and for $1 \le i < j \le |\sigma|$. The execution time of the trajectory

σ is total (σ) = $e_{|\sigma|}$.end – e_1.start. The trajectory space can be represented as ε^*.

Define 3 track prefixes: The track prefix $\sigma(k)$ is the first k events in the track σ, i.e., $\sigma^{(k)} = (e_1, \ldots, e_k)(1 \le k \le |\sigma|)$. For the trajectory prefix EEE, the remaining time for trajectory σ is *remain* (σ, k) = $e_{|\sigma|}$.end – e_k.end.

Define 4 process instances: A process instance is a complete execution of the entire business process and can be represented as a tuple $c = (cid, \sigma, p_1, \ldots, p_n)$. Where, $cid \in N^+$ is the ID, $\sigma \in E^*$ of the process instance, is the trajectory within the process instance, and for $1 \le i \le |\sigma|$, $\sigma(i) = cid$, where $\sigma(i)$ represents the *ith* event in the trajectory σ. Usually, $\sigma(i)$ is the start event of the process instance, and GGG is the end event of the process instance. $p_1 \in P_1, \ldots, p_n \in P_n$ is the attribute of the process instance, such as the executor and execution cost of the process instance.

Define 5 event logs: The event log is a collection of process instances, which records the historical execution of the business system, which can be represented as $L = \{\sigma_1, \ldots, \sigma|L|\}$.

3.2 Intelligent prediction of business process time

This paper uses a Redial basis function neural network (RBFNN) with an attention mechanism (Att-Bi-RBFNN) to build a remaining time prediction model. The model extends RBFNN in two aspects: first, a bidirectional structure is used to model the input trajectory prefix; second, the weight of each event in the trajectory prefix is automatically learned using an attention mechanism. **Figure 1** depicts the Att-Bi-RBFNN model's network architecture. The model primarily consists of the following important modules:

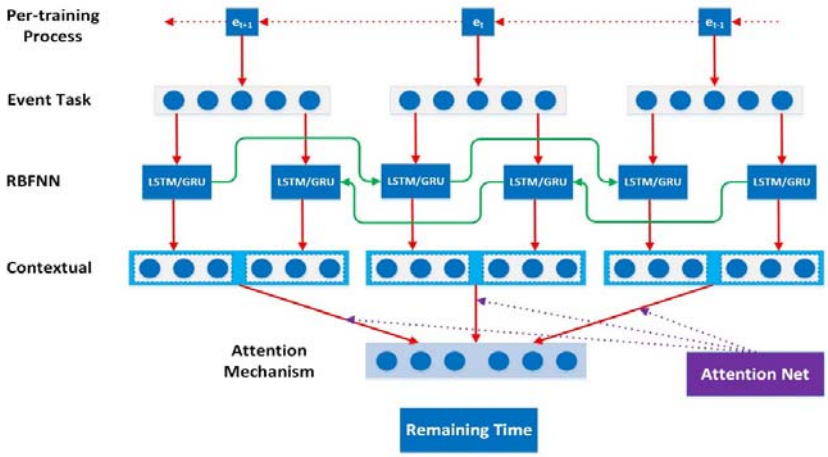

Figure 1. Recurrent neural networks based on attention mechanism.

1) Event representation

The model takes the trajectory $\sigma^{(k)} = (e_1, \ldots, e_k)$ as input and represents each event $e_t(1 \le t \le k)$ as the event vector GGG. Considering that an event is composed of activities, execution time and other attributes, for discrete attributes such as activities, this paper uses one-hot encoding to represent it as a 0/1 vector; properties are encoded in the same way. Finally, the event vector is obtained by concatenating the activity vector and the execution time vector. In the experiments in this paper, the event vector is used as part of the model parameters and is automatically updated during the training process.

2) Context encoding based on bidirectional RBFNN

This paper uses a bidirectional RBFNN to model trajectory prefixes. The bidirectional RBFNN consists of two traditional unidirectional RBFNN, which model the input trajectory from the forward and reverse perspectives, respectively. Compared with unidirectional RBFNN, bidirectional RBFNN can more comprehensively consider the correlation between events in the trajectory. The output latent vectors obtained by forward RBFNN and reverse RBFNN are denoted as $\underset{h_t}{\rightarrow}$ and $\underset{h_t}{\leftarrow}$, $\underset{h_t}{\rightarrow} = \text{RNN\_Model}(x_t, \underset{h_{t-1}}{\longrightarrow})$, $\underset{h_t}{\leftarrow} = \text{RNN\_Model}(x_t, \underset{h_{t+1}}{\longleftarrow})$, respectively. The context code at each moment is

obtained by concatenating the forward and reverse codes $h_t = [\overrightarrow{h_t}, \overleftarrow{h_t}]$. This paper is used two RBFNN implementations, LSTM and GRU.

3) Trajectory encoding based on attention mechanism

After obtaining the context encoding of each moment of the trajectory prefix, the encoding of the entire trajectory is further calculated as Equation (2).

$$v = \sum_{t=1}^{k} \alpha_t \cdot h_t \tag{2}$$

where, α_t is the weight of the context encoding at the t moment, which reflects the importance of the t-th event in the trajectory to the prediction of the remaining time to a certain extent. In this paper, a two-layer perceptron network (called Attention-Net) is used to calculate the context weight, as Equation (3).

$$\widetilde{\alpha_t} = \text{softmaz}(\text{g}^{\text{att}^{\text{T}}} \cdot \tanh(\text{W}^{\text{att}} \cdot h_t + \text{b}^{\text{att}}))$$
$$\alpha_t = \frac{\alpha_t}{\sum_{t=1}^{k} \widetilde{\alpha_t}} \tag{3}$$

where W^{att}, b^{att}, and g^{att} are the model parameters of the attention mechanism.

4) Remaining time prediction

According to the trajectory encoding, a fully connected network is used to build the remaining time prediction model. The specific calculation method is in Equation (4).

$$\text{remain} - \text{time}(\sigma^{(k)}) = g^T \cdot \text{ReLU}(W \cdot v + b) \tag{4}$$

where, W, b, and g are the model parameters of the fully connected network. This paper uses the Rectified Linear Unit (ReLU) as the activation function.

4. Result and discussion

These five datasets come from different fields. BPIC_2012_A/W/O records the loan application approval log of a financial institution; Helpdesk records the background log of a ticket management system; Hospital_Billing records the discharge settlement process log in a hospital ERP system. Statistics the information is shown in **Table 1**.

Table 1. Basic statistics of the dataset

Data set	Number of activities	Number of events	Number of tracks	Maximum track length	Minimum track length
BPIC_2012_A	10	73,192	13,276	9	3
BPIC_2012_W	5	147,230	9,478	148	1
BPIC_2012_O	6	41,728	5,032	36	4
Helpdesk	8	13,650	3,824	17	1
Hospital_billing	16	456,272	80,000	221	1

Statistical analysis should also be done on the training sets produced by these five event logs. **Figure 2** displays, as a function of track length, the number of tracks in the original event log and the matching training set. In addition, the solid line is the original event log statistics, and the dotted line is the statistics of the generated training set. It can be found that the number of trajectories in each event log basically decreases with the length, especially since the number of trajectories with long lengths is very rare, and so the number of trajectory prefixes in the generated training set also decreases sharply with the increase of length. These statistical results confirm the sparsity of the number of trajectory prefixes of specific lengths, especially longer lengths, and it is necessary to use transfer learning for model training.

The implications of each module in this strategy are broken down specifically below. First, the transfer learning training technique employed in this method is dropped, and then the conventional technique is utilized to train a single model for all length trajectory prefixes. The Att-Bi-RBFNN model's bidirectional module and attention module were then eliminated throughout the training phase, and LSTM and GRU, respectively, were tried as the RBFNN's particular implementations. **Figure 3** shows the MAE values of the above methods on various datasets. According to **Figure 3**, the following conclusions can be drawn:

First, compared to the regular LSTM and GRU, Bi-LSTM and Bi-GRU, respectively, attain lower MAE values with the inclusion of the bidirectional module; the addition of the attention module also results in lower MAE values in 3 and 4 of the 5 datasets.

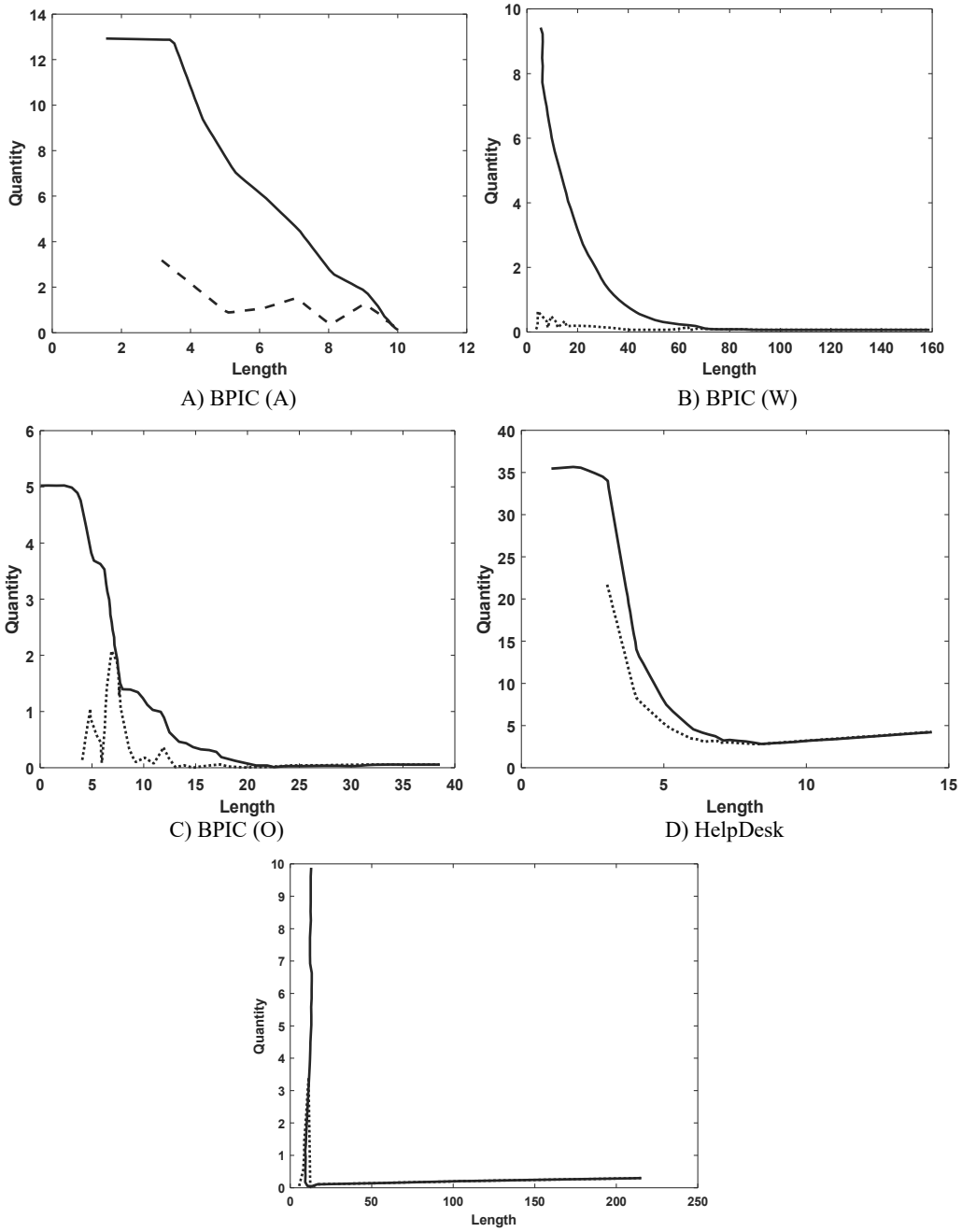

Figure 2. Show the statistical number of data set i) No. of tracks; ii) No. of trajectories.

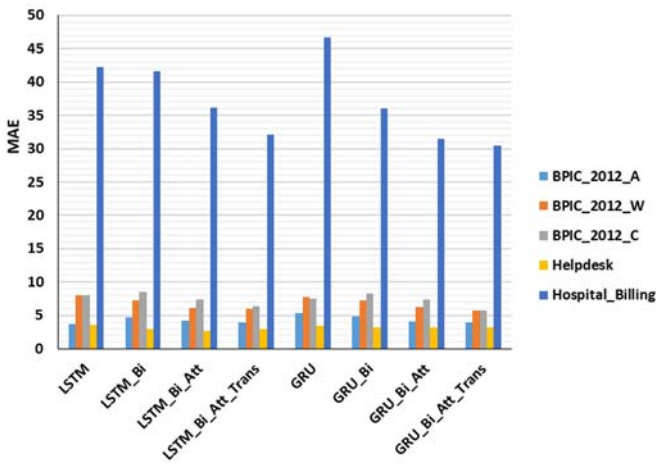

Figure 3. Final comparison of each model.

Then, compared to Bi-LSTM, Att-Bi-LSTM achieved lower MAE values in each of the five datasets, and compared to Bi-GRU, Att-Bi-GRU achieved lower MAE values in four of the five datasets. Overall, the addition of the bidirectional mechanism and the attention mechanism can enhance the RBFNN's performance on the job of remaining time prediction; However, the attention mechanism is particularly beneficial for enhancing the prediction effect.

Secondly, comparing Trans-Att-Bi-LSTM and Att-Bi-LSTM, Trans-Att-Bi-GRU and Att-Bi-GRU, the transfer learning model training method is more uniform than training on all datasets except Helpdesk. The model has a lower MAE value, which shows that using the transfer learning method to train the model separately for different length trajectory prefixes can greatly improve the accuracy of the remaining time prediction. When LSTM and GRU were put side by side, GRU outperformed LSTM on the average MAE value in 3 out of the 5 data sets. This indicates that GRU has certain advantages over LSTM when it comes to determining how much time is left in a business process.

The experiment compares the approach described in this research to the traditional approach based on the process model and the approach based on deep learning, as shown in **Table 2**.

Table 2. Comparison of the effects of benchmark methods

Method	BPIC (A)	BPIC (W)	BPIC (O)	Help desk	Hospital_billing
Proposed	4.438	6.821	6.863	4.299	33.187
SPN	8.88	8.516	6.385	6.337	78.018
LSTM	4.588	9.021	8.993	4.542	43.05
TS-set	6.505	7.429	6.392	7.283	52.456
Ts-multiset	6.488	7.691	6.203	7.167	52.507
TS-sequence	6.488	7.619	8.612	7.192	52.504

5. Conclusion

This study performs research on the estimation of remaining business process execution time using deep learning technology. This transfer learning technique and a more sophisticated deep neural network model, radial basis function neural network with attention mechanism, than previous studies that trained a single remaining time prediction model using the conventional LSTM network. The remaining time prediction models are trained separately for different trajectory lengths, which improves the pertinence of remaining time prediction. The findings of an experimental investigation that was conducted using five actual event logs show that the suggested strategy can drastically lower the MAE value of the remaining time forecast. Although the deep learning method used in this paper has a high prediction accuracy, it is difficult to interpret. Therefore, enhancing the method's interpretability in this publication is a crucial area for future research.

Conflict of interest

All authors have reported having no competing interests. The sponsors had not input into the study's conception or execution, data collection, analysis, or interpretation, article preparation, or decision to publish.

References

1. Teinemaa I, Dumas M, Rosa ML, *et al*. Outcome-oriented predictive process monitoring: Review and benchmark. ACM Transactions on Knowledge Discovery from Data (TKDD) 2019; 13(2): 1–57. doi: 10.1145/3301300.
2. der Aalst WMPV, Schonenberg MH, Song M. Time prediction based on process mining. Information Systems 2011; 36(2): 450–475. doi: 10.1016/j.is.2010.09.001.
3. Navarin N, Vincenzi B, Polato M, *et al*. LSTM networks for data-aware remaining time prediction of business process instances. In: Proceedings of 2017 IEEE Symposium Series on Computational Intelligence (SSCI); 2017 Nov 27–Dec 1; Honolulu. New York: IEEE; 2018. p. 1–7.
4. Senderovich A, Weidlich M, Gal A, Mandelbaum A. Queue mining for delay prediction in multi-class service processes. Information Systems 2015; 53:

278–295. doi: 10.1016/j.is.2015.03.010.
5. Rogge-Solti A, Weske M. Prediction of business process durations using non-Markovian stochastic Petri nets. Information Systems 2015; 54: 1–14. doi: 10.1016/j.is.2015.04.004.
6. Verenich I, Nguyen H, Rosa ML, *et al*. White-box prediction of process performance indicators via flow analysis. In: Proceedings of International Conference on Software and System Process; 2017 Jul 5–7; France. New York: Association for Computing Machinery; 2017. p. 85–94.
7. Tax N, Verenich I, Rosa ML, *et al*. Predictive business process monitoring with LSTM neural networks. In: Proceedings of International Conference on Advanced Information Systems Engineering; Jun 12–16; Essen. Berlin: Springer; 2017. p. 477–492.
8. Jimenez-Ramirez A, Barba I, Fernandez-Olivares J, *et al*. Time prediction on multi-perspective declarative business processes. Knowledge and Information Systems 2018; 57(3): 655–684. doi: 10.1007/s10115-018-1180-3.
9. Senderovich A, Di Francescomarino C, Ghidini C, *et al*. Intra and inter-case features in predictive process monitoring: A tale of two dimensions. In: Proceedings of International Conference on Business Process Management; 2017 Sept 10–15; Barcelona. Berlin: Springer; 2017. p. 306–323.
10. Folino F, Guarascio M, Pontieri L. Mining predictive process models out of low-level multidimensional logs. In: Proceedings of International Conference on Advanced Information Systems Engineering; 2014 Jun 16–20; Thessaloniki. Berlin: Springer; 2014. p. 533–547.
11. Jiménez-Ramírez A, Barba I, Del Valle C, *et al*. Generating multi-objective optimized business process enactment plans. In: Proceedings of International Conference on Advanced Information Systems Engineering; 2013 Jun 17–21; Valencia. Berlin: Springer; 2013. p. 99–115.
12. Rogge-Solti A, Mans RS, der Aalst WMPV, *et al.* Repairing event logs using timed process models. In: Proceedings of OTM Confederated International Conferences on the Move to Meaningful Internet Systems; 2013 Sept 9–13; Graz. Berlin: Springer; 2013. p. 705–708.
13. Polato M, Sperduti A, Burattin A, de Leoni M. Time and activity sequence prediction of business process instances. Computing 2018; 100(9): 1005–1031. doi: 10.1007/s00607-018-0593-x.
14. Verenich I, Dumas M, Rosa ML, *et al*. Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM Transactions on Intelligent Systems and Technology (TIST) 2019; 10(4): 1–34. doi: 10.1145/3331449.