Journal of Innovations in Computer Science and Engineering
Shahid Beheshti Unversity

January 2024, Volume 1, Issue 2

An Interpretable Method for Process Remaining Time Prediction based on Distance Metric Learning

Zahra Hosseini Nezhada, ORCID: 0009-0006-2612-6690
Sadegh Aliakbaryb✉, ORCID: 0000-0001-5773-1136
Ramak Ghavamizadeh Meibodic, ORCID: 0000-0002-1506-9380
Hamed Malek, ORCID: 0000-0003-4314-6539
aFaculty of computer science and engineering, Shahid Beheshti University, Tehran, Iran, email: z.hosseininezhad@sbu.ac.ir
bFaculty of computer science and engineering, Shahid Beheshti University, Tehran, Iran, email: s_aliakbary@sbu.ac.ir
cFaculty of computer science and engineering, Shahid Beheshti University, Tehran, Iran, email: R-Ghavami@sbu.ac.ir
dFaculty of computer science and engineering, Shahid Beheshti University, Tehran, Iran, email: h_malek@sbu.ac.ir

ABSTRACT

Process mining is a new research area that bridges the gap between data science and process science. Among the subfields of process mining, predictive process monitoring aims to predict process features such as the next event, outcome, and remaining time. In recent years, increasing research has been conducted in the area of remaining time prediction. In this paper, we present an interpretable method for remaining time prediction. Interpretability is an advantageous feature for a prediction method, because it provides the necessary advice and warnings to the organization's experts during the process execution when it is used in a recommender system. In the proposed method, we utilize distance metric learning methods to develop a distance function for process events. The distance function can be used to find the most similar cases to the intended case, and then the remaining time of the similar cases is used as the indicator of the remaining time of the intended case. Our proposed method has been evaluated on three datasets, and the evaluations revealed that it significantly outperforms the state-of-the-art baseline in two datasets while achieving comparable accuracy in the third dataset. Additionally, our proposed method provides interpretation, while the competitor methods are not interpretable.

KEYWORDS:

Business Process, Process Mining, Remaining Time Prediction, Distance Metric, Interpretable Prediction.

1. INTRODUCTION

Process mining is part of data science and process science, and it can be considered the bridge between them. Process mining techniques use machine learning, process modeling, and analysis to extract valuable knowledge from the event log [1]. One of the concerns of organizations is the ability to predict the future. Organizations are constantly monitoring business processes to be aware of the current situation; now, if they can predict the future state of the process, they will make better decisions. Predictive business processes monitoring is a subfield of process mining that belongs to process enhancement field, and it responds to this essential requirement of organizations [2].

Predictive business process monitoring techniques are used to forecast the future value for a target process instance [3, 4]. For this purpose, the event log is used as the input, and the output is a predicted value such as the next activity or sequence of activities [5, 6], the possible outcome of a running case [7, 8], or the remaining time of a case until the end [9, 10, 11, 12]. These predictions help professionals make decisions. They also provide alerts about process features such as cycle time or remaining time [4]. For example, there may be a time limitation for executing a process, so we can detect the delay sooner by predicting the remaining time and considering a resource allocation solution.

In recent years, researchers have proposed different approaches for predicting the remaining time of the business processes. Remaining time prediction is the task of predicting the time of an unfinished case until its completion. This prediction have many benefits and applications, for example, to predict the violation or non-violation of the service level agreement, or to help business owners to make decisions for dealing with possible delays [4]. In order to measure the accuracy of the remaining time prediction methods, metrics such as Mean Absolute Error (MAE) and Root Mean Square Error (RMSE) are suitable since we encounter a continuous variable [3]. Although most of the existing methods have focused on optimizing the accuracy of the prediction function, the interpretability of the prediction method is also essential. In this study, we have also paid attention to the interpretability of the method along with its accuracy. By interpretability, we mean that we can describe the reason behind the

Submit Date: 2023-12-24
Revise Date: 2024-05-28
Accept Date: 2024-06-15
✉ Corresponding author

predicted values and answer the "why" question. Our method is considered interpretable because we develop a distance function that finds the most similar cases to the target case, and therefore, our prediction is interpreted based on the most similar cases.

Our main contribution is a solution to the remaining time prediction problem, which not only improves the accuracy of the prediction but also enables the interpretation of the predictions. The rest of this paper is structured as follows: Section 2 summarizes the related works. Section 3 explains the proposed approach in detail. Section 4 presents the datasets and compares the proposed approach with other state-of-the-art methods. Finally, Section 5 concludes the proposed approach and discusses the direction of future work.

2. RELATED WORKS

One of the predictive process monitoring problems is the "remaining time prediction", which is addressed in many researches. Different algorithms have been introduced in the literature to predict the remaining time of business processes [4]. In some approaches, the process model is used for prediction, where such methods are called process-aware [3]. For example, Rogge-Solti and Weske have used a stochastic Petri net model to predict the remaining service execution time [9]. They have extended this approach in [10] using the elapsed time to achieve higher accuracy. Some process-aware methods have utilized transition systems as the process model [11]. In this approach, the transition system is created from the event log. The process model should be created to balance overfitting and underfitting [13], and then, this model is used to predict the process completion time [11].

Among different utilized techniques, machine learning is one of the most used and popular approaches to solving the "remaining time prediction problem". In this problem, we face a regression task, so regression methods are used [4, 14, 15]. Some researchers have considered Intra-case (information about the running case) and Inter-case (information about the execution of other cases) features to reduce prediction error [16, 17]. For example, Senderovich et al. [16] have used XGBoost, and L. Klijn et al. [17] have used two supervised machine learning algorithms: random forest and XGBoost. In some studies, the cases are clustered according to control flow information, and a predictive model is trained for each cluster. Ogunbiyi et al. [18] have provided a comparative analysis of five clustering-based approaches to predict the remaining time and found the approach that clusters traces based on activities outperforms other approaches.

Each approach has some advantages and disadvantages; therefore, using hybrid techniques can be helpful. For example, Ni et al. [19] have presented a method using the auto-encoded transition system to predict the remaining time. Aburomman et al. [20, 21] have defined the number of attributes on the event log, extended the annotated transition system (ATS) model to include these features, and utilized a linear regression for remaining time prediction.

Recently, artificial neural networks and deep learning methods have become more popular [2]. Recurrent Neural Networks (RNN) are applicable for remaining time prediction because they are suitable for sequential data such as event log, and they can remember the dependencies between the data. LSTM [22, 23, 24] is one type of RNNs that is widely have utilized in the field of process mining [2] and remaining time prediction [4]. Recent experiments have showed that the state-of-the-art methods in remaining time prediction are based on LSTM neural networks [4]. On the other hand, artificial neural networks have some drawbacks, such as the need for high computational resources and the lack of interpretability [4]. Therefore, one challenge in this area is introducing an accurate and interpretable method.

Interpretability and explainability are emerging concerns in artificial intelligence (AI); therefore, in process mining, some recent researchers have considered interpretable models [25, 26, 27, 28, 29]. Stierle et al. [28] have identified techniques for explainable predictive business process monitoring (XPBPM), and according to their studies, the terms "explainability" and "interpretability" are used interchangeably in this research area. Williams et al. [25] have used posthoc explainers such as LIME or SHAP to show why a predictive model makes a mistake and tries to improve its accuracy. Galanti et al. [26] have sought to provide an explainable framework for predictive process monitoring by using the game theory of Shapley Values, independent of the technique used to make a prediction. Wickramanayake et al. [27] have proposed event attention (the effect of a specific event on prediction) and attribute attention (the effect of each attribute of the event on prediction). They also have considered two attention mechanisms, including shared and specialised, and finally, Wickramanayake et al. [27] have built two attention-based interpretable models. Cao et al. [30] have proposed a gated RNN using a reachability graph to improve the performance of remaining time prediction and explain the remaining time prediction model.

3. PROPOSED APPROACH

In this section, we introduce an approach to predicting the remaining time of an ongoing process case. The proposed method receives the target process case along with the event log of different process cases as input and predicts the remaining time of the target case as the output.

The main idea of our proposed method is to look for the most similar cases to the target case. We hypothesize that if two cases are similar, it is likely that their remaining time will be similar. To verify this hypothesis, defining a distance function for cases is necessary. The distance function gets two cases and returns a real value as the dissimilarity between the two events. We should first convert each case into a feature vector of real number values to utilize the distance metrics for comparing the cases. Then, we may use typical distance metrics such as Manhattan distance or Euclidean distance to compare cases. However, our experiments showed that the accuracy of such naive distance metrics is not acceptable when employed in the remaining time prediction methods. Therefore, better distance functions are needed for this application. To this aim, we utilized machine learning methods for distance metric learning to define a suitable distance function, which resulted in an accurate prediction method.

It is also worth noting that using the distance function not only results in high prediction accuracy but also makes the proposed method interpretable because it explains and supports the predicted values with evidence of similar cases. In other words, we can answer questions like "Why does the prediction method result in such estimation for the remaining time of the target case?" by presenting the most similar cases.

3.1. The Architecture of the Proposed Method

An overview of the proposed approach is presented in Figure 1. In the following, we will discuss different components of our method in more detail. Our proposed method has three main steps: "Data Preprocessing", "Distance Function Learning", and "Finding Similar Cases". Like other predictive monitoring methods [4], our method has two phases: offline and online. In the offline phase, we preprocess data and train a model for creating the distance function, and in the online phase, we use the distance function to predict the remaining time and provide some interpretation. The following illustrates these steps more.

The "Data Preprocessing" component, the first step in our method, takes the event log as input and creates two datasets that are crucial for the subsequent steps. To ensure a clear understanding of the structure of these datasets, we first define some key terms formally.

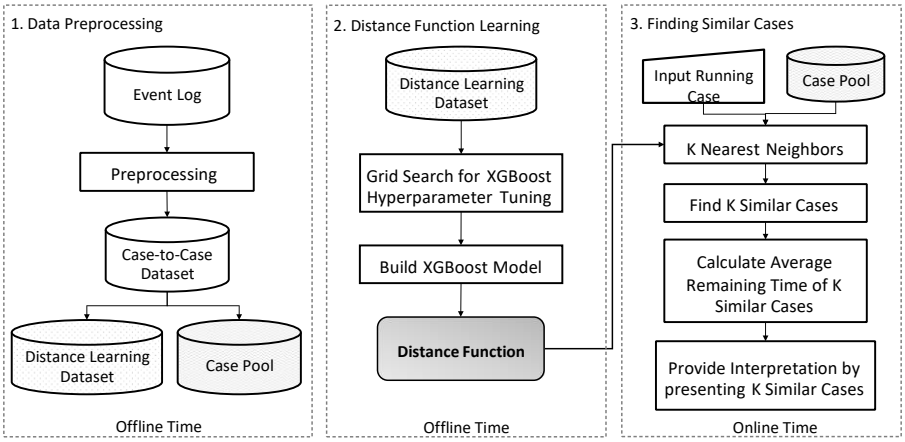


Figure 1 The architecture of the proposed method

Let A stand for the set of possible process activities, C for the set of cases, E for the universe of events, T for the time, and D for the set of attributes such as resource. Then, each case is a running process instance and contains several events. Each event e is defined in definition (1), in which a is the activity name, c is the case identifier, t is the timestamp, and d_i is an attribute.

In definition (2), π_T is a function that maps the event to its timestamp, and in definition (3), π_C is a function that maps the event to its related case [2]. In definition (4), trace is a non-empty sequence of events that refers to the same case and event log is a set of completed traces [4].

In the "Data Preprocessing" step, a new case-to-case (C2C) dataset is created. The rows of the C2C dataset are described in the definition (5). We have put each pair of rows of the event log (e_i and e_j) side by side, along with their amount of dissimilarity in the C2C dataset. The dissimilarity between two cases, which is described in definition (6), is simply considered as the absolute difference between their remaining time, which is specified in definition (7). The reason for choosing such a dissimilarity measure is that we assume that the more similar the two cases are, the shorter the distance between their remaining time.

$$e = (a, c, t, d_i), a \in A, c \in C, t \in T, d_i \in \{D_i\}, i \in [1, m] \quad (1)$$

$$\pi_T(e) = t, e \in E \quad (2)$$

$$\pi_C(e) = c, e \in E \quad (3)$$

$$\sigma = \langle e_1, \ldots, e_n \rangle, \forall i, j \in [1, n]: \pi_C(e_i) = \pi_C(e_j) \quad (4)$$

$$C2C = \{(e_i, e_j, dissimilarity(e_i, e_j))\}, e_i, e_j \in E \quad (5)$$

$$dissimilarity(e_i, e_j) = \left|RT(e_i) - RT\left(e_j\right)\right|, e_i, e_j \in E \quad (6)$$

$$RT(e) = \pi_T(e_n) - \pi_T(e) \quad (7)$$

As mentioned earlier, the input is an incomplete running instance of the process (called case). To train a machine learning model, all features must be encoded as a fixed-width feature vector. Various encoding methods have been presented in the literature [4]. For example, one of the encoding techniques is the last state encoding, where only the last m events of the current case are considered (m=1 is the most common) [4]. As it can be seen in (5), in our proposed approach, last state encoding with m=1 is used due to time and space limitations, but it is obvious that by considering a higher value of m, we use more information from the running case. For example, if m=2, two last events of the running case are encoded; therefore, we measure the impact of considering the two last events in Section 4.

Attributes can be numerical or categorical and most of machine learning methods are designed to deal with numerical values. We normalize numerical values and for categorical values, we used one-hot encoding to convert them into numerical data. Then we split the C2C dataset into distance learning dataset and case pool by putting 80% of the dataset for distance learning and 20% for case pool. We used distance learning dataset for "Distance Function Learning" and the case pool for "Finding Similar Cases", as described in Figure 1. Separating these two datasets is necessary because we are going to use the distance function in the third step and should not test it on the part of the data that has already been used in creating the distance function.

As Figure 1 shows, the second step is "Distance Function Learning". In this step, we use the distance learning dataset, then we utilize machine learning methods to create a distance function. We employed extreme gradient boosting (XGBoost) method [31] as a regression algorithm which predicts the distance between two considered cases using the distance learning dataset. XGBoost stands for Extreme Gradient Boosting and it is a scalable tree boosting with advantages such as more accuracy, fewer resources and etc [31]. XGBoos has several hyperparameters, which we tune using grid search. The final output of this step is a distance function, which we will utilize in the next step.

After developing a distance function, in the last step, "Finding Similar Cases", we employ the distance metric in the k nearest neighbors method for estimating the remaining time of the target process case. In this regard, for each input case, we first find the k most similar cases in the case pool using the trained distance function in the previous step. Then, we calculate the average remaining time of the k most similar cases. This value is the predicted value for the remaining time of the input case.

The intuition of this idea is that the more similar the two cases, the more similar the remaining time. Moreover, in terms of interpretability, we can provide a reason for whatever value we report here by presenting the most similar cases to determine the reason for the prediction. While other previous studies do not focus on the relationship between this case and other cases, they only report a single value, and the user will not know how and why this value is calculated.

4. EVALUATION

This section presents the experiments and the evaluation results. We first describe the utilized datasets, and then, we explain the evaluation criteria and the baseline methods. Finally, the evaluation results are presented.

4.1. Implementation Details

For implementation, we used Python and its libraries, where the details are presented in Table 1.

Table 1 Implementation Details

Title	Version	Description
Python	3.7.3	Programming language
Scikit-learn	0.23.2	Python library for machine learning applications
Pandas	0.24.2	Python library for data preprocessing and analysis
Numpy	1.16.2	Python library for working with large, multi-dimensional arrays and matrices
XGboost	1.2.1	Python library which provides gradient boosting

We used a grid search to optimize hyperparameters in the "Distance Function Learning" step to achieve an accurate prediction (see Figure 1). Table 2 lists the most sensitive learning hyperparameters we optimized during the grid search.

Table 2 The tuned values of hyper-parameters according to the grid search approach.

Hyperparameter	Tuned value
n-estimator	500
learning-rate	0.06
subsample	0.8
colsample-by tree	0.8
max-depth	6
k	{1,2,3,4,5}

4.2. Dataset

Table 3 describes the datasets we used to evaluate the proposed method. The dataset is described from different points of view, including the number of cases, number of events, number of activity types, and number of attributes by type. The Helpdesk [32] belongs to the help desk log of an Italian Company and contains some information about the ticketing management system. The BPIC 2012 [33] is an abbreviation for Business Process Intelligence Challenge 2012, and its domain is a loan application process. Finally, the Production [34] is about a production process. These datasets are publicly available at the 4TU Center for Research Data[1].

[1] https://data.4tu.nl/search?search="event+log"

Table 3 Dataset Description

Dataset	Domain	# Cases	# Events
Helpdesk [32]	Customer Service	3218	10598
BPIC 2012 [33]	Financial	3487	15908
Production [34]	Manufacturing	225	4543

4.3. Evaluation Criteria

We consider the "mean absolute error" (MAE) as the evaluation criteria of the proposed method, which is described in (8), in which y_i shows the actual remaining time of the case i and $\acute{y}_i$ shows the predicted value.

$$MAE = \frac{1}{N}\sum_{i=1}^{N} | y_i - \acute{y}_i | \quad (8)$$

4.4. Baseline Methods

Table 4 describes the state-of-the-art methods which we have used as the baselines for evaluating the proposed method. We have considered a wide variety of method categories among the available solutions in the literature, and for each category, we selected the most accurate method.

Table 4 Baseline methods

Method	Acronym	Approach
Transition System [11]	TS	Process-aware approach
Artificial Neural Networks [22]	LSTM	Deep neural networks
Stochastic Petri nets [10]	SPN	Process-aware approach
Flow Analysis [35]	FA	Hybrid method (regression + classification + flow analysis)
Regression Learning [4]	XGBoost	Last state encoding, Zero bucketing

4.5. Evaluation Results

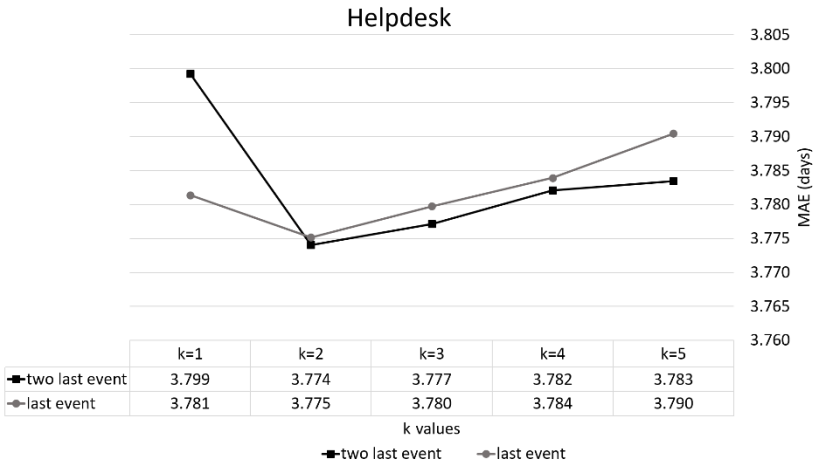
In this section, we want to analyze the evaluation results. We evaluated different parameters; one of them is k. It is an important parameter that is effective in our proposed method. Therefore, we measure the MAE for different k values for each dataset to measure its effect.

Another idea presented in our proposed method is to consider different values of m when using last state coding. In the C2C dataset (see in (5)), we used last state encoding with m=1, which means we only considered the last event of the case. In this section, we also evaluate m=2, but due to space and time limitations, only the activity of the event before the last event is used. The intuition behind this idea is that the similarity of the previous activities to the last event indicates the similarity of the two cases.

Figure 2 shows the effect of considering different k parameter values and considering or not considering the previous activity of the last event on different datasets with our proposed method. In Figure 2, the vertical axis is the MAE in days, and the horizontal axis is different values of k in the range of 1 to 5. Two charts are drawn for each dataset; in the gray chart, only the last event is considered (see definition (5)), and in the gray chart, two last events is considered. In the following, we will review the results for each dataset in detail.

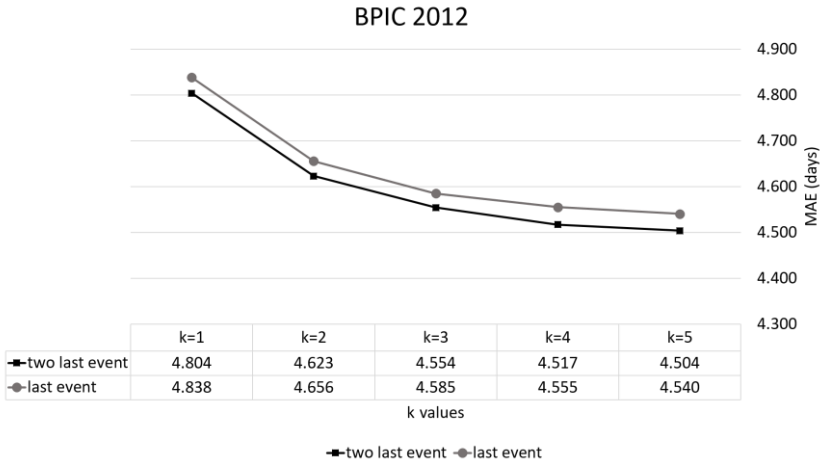
According to Figure 2a in Helpdesk, the best value for k is 2, and considering the two last events does not have a significant effect. However, increasing k and considering the two last events reduced MAE slightly. In BPIC (Figure 2b), the MAE decreased with increasing k, and considering two last events. Using the two last events reduced the MAE in Production (Figure 2c), but by increasing the value of k, this reduction is less noticeable, and for this dataset, the best value of k is 3.

Figure 2 shows that considering two last events usually reduces the MAE. The value of k is different based on the dataset, but k>1 is a better choice in all three datasets. When considering k=1, we choose the remaining time equal to the most similar case. Therefore, if there is noise or an outlier, MAE will increase, and by averaging over k similar cases, the effect of noise and outliers is reduced.



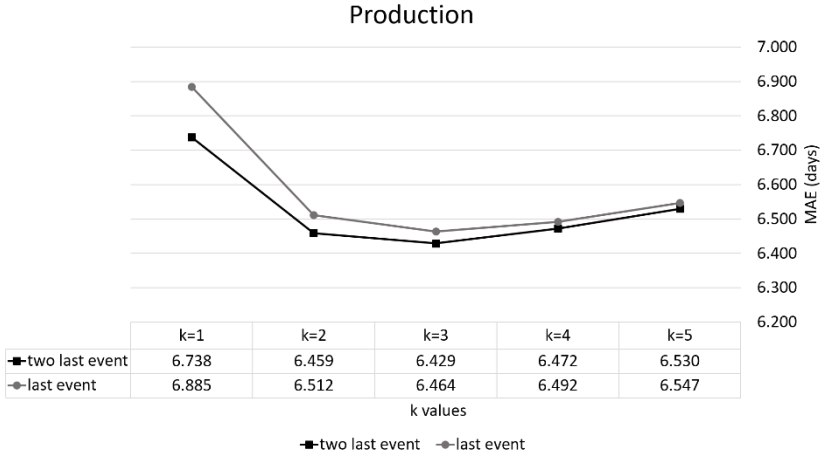
	k=1	k=2	k=3	k=4	k=5
two last event	3.799	3.774	3.777	3.782	3.783
last event	3.781	3.775	3.780	3.784	3.790

a) Helpdesk dataset



	k=1	k=2	k=3	k=4	k=5
two last event	4.804	4.623	4.554	4.517	4.504
last event	4.838	4.656	4.585	4.555	4.540

b) BPIC2012 dataset

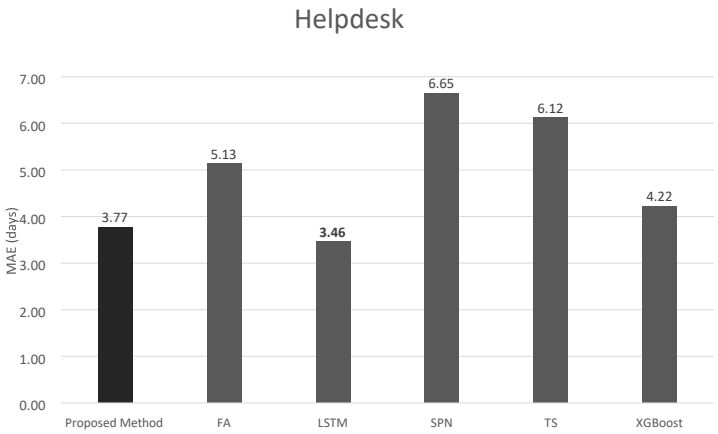


	k=1	k=2	k=3	k=4	k=5
two last event	6.738	6.459	6.429	6.472	6.530
last event	6.885	6.512	6.464	6.492	6.547

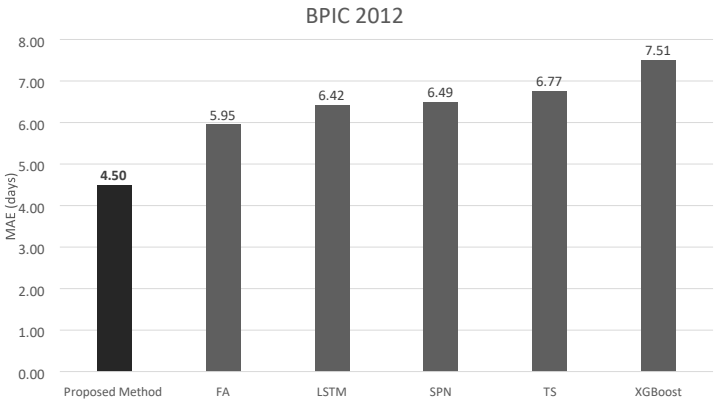
c) Production dataset

Figure 2 The MAE for the proposed method according to different k values and different m values in last state encoding

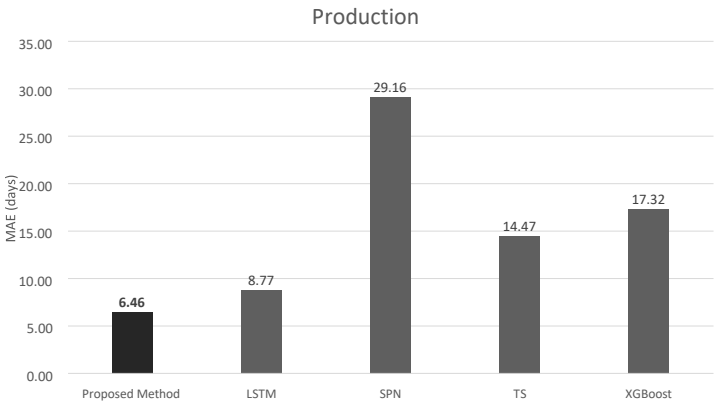
Figure 3 compares the proposed method results on each dataset (by considering best k, m) with other baseline methods (reproduced by I. Verenich et al. [4]). Our proposed method can achieve higher accuracy than the state-of-the-art baseline in two datasets and slightly lower in one dataset. In general, our proposed method has acceptable accuracy and high interpretability.



(a) Helpdesk dataset



(b) BPIC2012 dataset



(c) Production dataset

Figure 3: MAE in days over all methods

5. CONCLUSION AND FUTURE WORKS

In this study, we investigated the issue of predicting the remaining time in business processes. According to existing methods, LSTM neural network has become more accurate on most datasets [4], and today most existing research is looking for a way to outperform LSTM and solve its challenges like interpretability. In our proposed method, by changing the view of the problem, we seek to find the most similar case to the current case. The advantage of this method is its high interpretability and simplicity. To find the most similar case, we used a machine learning model to learn the distance metric function. After obtaining the most similar cases, we used k most similar cases and took their averaging remaining time as the remaining time of the current case.

This proposed method, with advantages such as acceptable accuracy and high interpretability, has time-consuming and space-consuming issues. There are some ideas for solving them. For example, we can use a heuristic function when finding a similar case. Another idea is to use clustering and compare the current case with only the cluster centroid. In general, hybrid methods can solve the problems of time-space consumption. In addition, paying attention to the process model can effectively improve the proposed method's accuracy, which we will address in future works.

6. REFERENCES

[1] W. van der Aalst, Process Mining, Springer Berlin Heidelberg, 2016. doi: 10.1007/978-3-662-49851-4. URL https://doi.org/10.1007/978-3-662-49851-4

[2] E. Rama-Maneiro, J. Vidal, M. Lama, Deep learning for predictive business process monitoring: Review and benchmark, IEEE Transactions on Services Computing (2022) 1–1doi:10.1109/tsc.2021.3139807. URL https://doi.org/10.1109/tsc.2021.3139807

[3] E. M´arquez-Chamorro, M. Resinas, A. Ruiz-Cort´es, Predictive monitoring of business processes: A survey, IEEE Transactions on Services Computing 11 (6) (2018) 962–977. doi:10.1109/TSC.2017.2772256.

[4] Verenich, M. Dumas, M. L. Rosa, F. M. Maggi, I. Teinemaa, Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring, ACM Trans. Intell. Syst. Technol. 10 (4) (Jul. 2019). doi:10.1145/3331449. URL https://doi.org/10.1145/3331449

[5] J. Evermann, J.-R. Rehse, P. Fettke, A deep learning approach for predicting process behaviour at runtime, in: Business Process Management Workshops, Springer International Publishing, 2017, pp. 327–338. doi: 10.1007/978-3-319-58457-7_24. URL https://doi.org/10.1007/978-3-319-58457-7_24

[6] S. van der Spoel, M. van Keulen, C. Amrit, Process prediction in noisy data sets: A case study in a dutch hospital, in: Lecture Notes in Business Information Processing, Springer Berlin Heidelberg, 2013, pp. 60–83. doi: 10.1007/978-3-642-40919-6_4. URL https://doi.org/10.1007/978-3-642-40919-6_4

[7] Leontjeva, R. Conforti, C. D. Francescomarino, M. Dumas, F. M. Maggi, Complex symbolic sequence encodings for predictive monitoring of business processes, in: Lecture Notes in Computer Science, Springer International Publishing, 2015, pp. 297–313. doi:10.1007/978-3-319-23063-4_21. URL https://doi.org/10.1007/978-3-319-23063-4_21

[8] F. M. Maggi, C. D. Francescomarino, M. Dumas, C. Ghidini, Predictive monitoring of business processes, in: Advanced Information Systems Engineering, Springer International Publishing, 2014, pp. 457–472. doi:10.1007/978-3-319-07881-6_31. URL https://doi.org/10.1007/978-3-319-07881-6_31

[9] Rogge-Solti, M. Weske, Prediction of remaining service execution time using stochastic petri nets with arbitrary firing delays, in: Service-Oriented Computing, Springer Berlin Heidelberg, 2013, pp. 389–403. doi:10.1007/978-3-642-45005-1_27. URL https://doi.org/10.1007/978-3-642-45005-1_27

[10] Rogge-Solti, M. Weske, Prediction of business process durations using non-markovian stochastic petri nets, Information Systems 54 (2015) 1 –14. doi:https://doi.org/10.1016/j.is.2015.04.004. URL http://www.sciencedirect.com/science/article/pii/ S0306437915000642

[11] W. van der Aalst, M. Schonenberg, M. Song, Time prediction based on process mining, Information Systems 36 (2) (2011) 450 – 475, special Issue: Semantic Integration of Data, Multimedia, and Services. doi:https://doi.org/10.1016/j.is.2010.09.001. URL http://www.sciencedirect.com/science/article/pii/ S0306437910000864

[12] F. van Dongen, R. A. Crooy, W. M. P. van der Aalst, Cycle time prediction: When will this case finally be finished?, in: On the Move to Meaningful Internet Systems: OTM 2008, Springer Berlin Heidelberg, 2008, pp.319–336. doi:10.1007/978-3-540-88871-0_22. URL https://doi.org/10.1007/978-3-540-88871-0_22

[13] W. M. Van der Aalst, V. Rubin, H. Verbeek, B. F. van Dongen, E. Kindler, C. W. Gu¨nther, Process mining: a two-step approach to balance between underfitting and overfitting, Software & Systems Modeling 9 (1) (2010) 87–111.

[14] M. de Leoni, W. M. P. van der Aalst, M. Dees, A general framework for correlating business process characteristics, in: S. Sadiq, P. Soffer, H. V¨olzer (Eds.), Business Process Management, Springer International Publishing, Cham, 2014, pp. 250–266.

[15] M. de Leoni, W. M. van der Aalst, M. Dees, A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs, Information Systems 56 (2016) 235 – 257. doi:https://doi.org/10.1016/j.is.2015.07.003. URL http://www.sciencedirect.com/science/article/pii/ S0306437915001313

[16] Senderovich, C. Di Francescomarino, C. Ghidini, K. Jorbina, F. M. Maggi, Intra and inter-case features in predictive process monitoring: A tale of two dimensions, in: J. Carmona, G. Engels, A. Kumar (Eds.), Business Process Management, Springer International Publishing, Cham, 2017, pp.306–323.

[17] E. L. Klijn, D. Fahland, Identifying and reducing errors in remaining time prediction due to inter-case dynamics, in: 2020 2nd International Conference on Process Mining (ICPM), IEEE, 2020. doi:10.1109/icpm49681.2020.00015. URL https://doi.org/10.1109/icpm49681.2020.00015

[18] N. Ogunbiyi, A. Basukoski, T. Chaussalet, Comparative analysis of clustering-based remaining-time predictive process monitoring approaches, International Journal of Business Process Integration and Management 10 (3-4) (2021) 230–241.

[19] W. Ni, M. Yan, T. Liu, Q. Zeng, Predicting remaining execution time of business process instances via auto-encoded transition system, Intelligent Data Analysis 26 (2) (2022) 543–562. doi:10.3233/ida-215755. URL https://doi.org/10.3233/ida-215755

[20] Aburomman, M. Lama, A. Bugar´ın, A vector-based classification approach for remaining time prediction in business processes, IEEE Access 7 (2019) 128198–128212.

[21] Aburomman, M. Lama, A. Bugar´ın-Diz, Estimating remaining time of business processes with structural attributes of the traces, in: Computational Intelligence and Mathematics for Tackling Complex Problems 2, Springer, 2022, pp. 47–54.

[22] N. Navarin, B. Vincenzi, M. Polato, A. Sperduti, Lstm networks for dataaware remaining time prediction of business process instances, in: 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 2017, pp. 1–7. doi:10.1109/SSCI.2017.8285184.
[23] N. Tax, I. Verenich, M. L. Rosa, M. Dumas, Predictive business process monitoring with LSTM neural networks, in: Advanced Information Systems Engineering, Springer International Publishing, 2017, pp. 477–492. doi:10.1007/978-3-319-59536-8_30. URL https://doi.org/10.1007/978-3-319-59536-8_30
[24] R. Cao, W. Ni, Q. Zeng, F. Lu, C. Liu, H. Duan, Remaining time prediction for business processes with concurrency based on log representation, China Communications 18 (11) (2021) 76–91.
[25] W. Rizzi, C. Di Francescomarino, F. M. Maggi, Explainability in predictive process monitoring: When understanding helps improving, in: D. Fahland, C. Ghidini, J. Becker, M. Dumas (Eds.), Business Process Management Forum, Springer International Publishing, Cham, 2020, pp. 141–158.
[26] R. Galanti, B. Coma-Puig, M. de Leoni, J. Carmona, N. Navarin, Explainable predictive process monitoring, in: 2020 2nd International Conference on Process Mining (ICPM), IEEE, 2020. doi:10.1109/icpm49681.2020.00012. URL https://doi.org/10.1109/icpm49681.2020.00012
[27] Wickramanayake, Z. He, C. Ouyang, C. Moreira, Y. Xu, R. Sind-hgatta, Building interpretable models for business process prediction using shared and specialised attention mechanisms, Knowledge-Based Systems248 (2022) 108773. doi:10.1016/j.knosys.2022.108773. URL https://doi.org/10.1016/j.knosys.2022.108773
[28] M. Stierle, J. Brunk, S. Weinzierl, S. Zilker, M. Matzner, J. Becker, Bringing light into the darkness-a systematic literature review on explainable predictive business process monitoring techniques (2021).
[29] N. Mehdiyev, M. Majlatow, P. Fettke, Interpretable and explainable machine learning methods for predictive process monitoring: A systematic literature review, arXiv preprint arXiv:2312.17584 (2023).
[30] R. Cao, Q. Zeng, W. Ni, H. Duan, C. Liu, F. Lu, Z. Zhao, Business process remaining time prediction using explainable reachability graph from gated rnns, Applied Intelligence 53 (11) (2023) 13178–13191.
[31] T. Chen, C. Guestrin, XGBoost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, New York, NY, USA, 2016.
[32] M. Polato, Dataset belonging to the help desk log of an italian company (2017). doi:10.4121/UUID:0C60EDF1-6F83-4E75-9367-4C63B3E9D5BB. URL https://data.4tu.nl/articles/_/12675977/1
[33] B. van Dongen, Bpi challenge 2012 (2012). doi:10.4121/UUID:3926DB30-F712-4394-AEBC-75976070E91F. URL https://data.4tu.nl/articles/_/12689204/1
[34] D. Levy, Production analysis with process mining technology (2014). doi:10.4121/UUID:68726926-5AC5-4FAB-B873-EE76EA412399. URL https://data.4tu.nl/articles/_/12697997/1
[35] I. Verenich, M. Dumas, M. La Rosa, H. Nguyen, Predicting process performance: A white-box approach based on process models, Journal of Software: Evolution and Process 31 (6) (2019) e2170.

Zahra Hosseini Nezhad is a PhD candidate of Computer Engineering at Shahid Beheshti University, Tehran, Iran. Her research interests include process mining, predictive business process, interpretable AI.

Sadegh Aliakbary is an assistant professor in Software and Information Systems at Shahid Beheshti University, Tehran, Iran. His research interests include social network analysis, data mining, software architecture and software quality assurance.

Ramak Ghavamizadeh Meybodi is an assistant professor in Software and Information Systems at Shahid Beheshti University, Tehran, Iran. Her research interests include algorithms, graph and social network.

Hamed Malek is an assistant professor in Artificial Intelligence at Shahid Beheshti University, Tehran, Iran. His research interests include deep learning, big data, and bio-computing.