2020 IEEE 24th International Enterprise Distributed Object Computing Conference (EDOC)

Attention Mechanism in Predictive Business Process Monitoring

1st Abdulrahman Jalayer
Ferdowsi University of Mashhad
Mashhad, Iran
rahman.jalayer@mail.um.ac.ir

2st Mohsen Kahani
Ferdowsi University of Mashhad
Mashhad, Iran
kahani@um.ac.ir

3st Amin Beheshti
Macquarie University
Sydney, Australia
amin.beheshti@mq.edu.au

4st Asef Pourmasoumi
Ferdowsi University
Mashhad, Iran
a.pms@um.ac.com

5st Hamid Reza Motahari-Nezhad
EY AI Lab
Palo Alto, USA
hamid.motahari@ey.com

***Abstract*—Business process monitoring techniques have been investigated in depth over the last decade to enable organizations to deliver process insight. Recently, a new stream of work in predictive business process monitoring leveraged deep learning techniques to unlock the potential business value locked in process execution event logs. These works use Recurrent Neural Networks, such as Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU), and suffer from misinformation and accuracy as they use the last hidden state (as the context vector) for the purpose of predicting the next event. On the other hand, in operational processes, traces may be very long, which makes the above methods inappropriate for analyzing them. In addition, in predicting the next events in a running case, some of the previous events should be given a higher priority. To address these shortcomings, in this paper, we present a novel approach inspired by the notion of attention mechanism, utilized in Natural Language Processing and, particularly, in Neural Machine Translation. Our proposed approach uses all hidden states to accurately predict future behavior and the outcome of individual activities. Experimental evaluation of real-world event logs revealed that the use of attention mechanisms in the proposed approach leads to a more accurate prediction.**

***Index Terms*—Business Process Management, Process Mining, Predictive Process Monitoring, Attention Mechanism, Deep Learning, LSTM, Seq2Seq**

I. Introduction

A business process (BP) is a set of coordinated tasks and activities, carried out manually or automatically, to achieve a business objective or goal‘ [1], [2]. Business Process Management (BPM), i.e., a generic software system that is driven by explicit process designs to enact and manage operational BPs, enable organizations to be more efficient and capable of process automation throughout the process management life cycle [3]–[6]. Process mining is an emerging discipline that bridges the gap between traditional BPM and data-centric analysis techniques such as machine learning and data mining [1], [7]–[9]. A major task in process mining is process monitoring with the aim of detecting the undesired deviations occurring during the process execution, which makes it possible to oversee the whole process. Process monitoring can be offline or online [10]. The offline process monitoring is realized by traditional monitoring methods. The idea is to give, as the input, a dataset including completed process instances and receive such outputs as the deviations that have occurred [10].

An undesired deviation sometimes inflicts irreparable costs and wasted time in an organization. This makes it necessary to identify and make alarming notifications about deviations before they happen. This will provide the opportunity to take preventive measures [10]. Deviations are of a variety of types such as undesired deviations from the desired workflow or deadline violations [10]. In the case of deadline violations, for instance, if predicted accurately, it can save time and cost for organizations. Predictive business process monitoring techniques deal with the prediction of future sequences in a running case by using historical information extracted from event logs [11]. Future prediction using historical data is shown schematically in Figure 1. As seen in Figure 1, the event log is first split randomly into two sets of training and testing, where T_k denotes the k-th trace and a_1^{k+1} is the first activity of the $(k+1)$-th trace. The model is then learned on the training data using a learning engine such as LSTM [12]. The output of this phase would be a trained model. Then, in the testing phase, when we aim to predict the future behavior

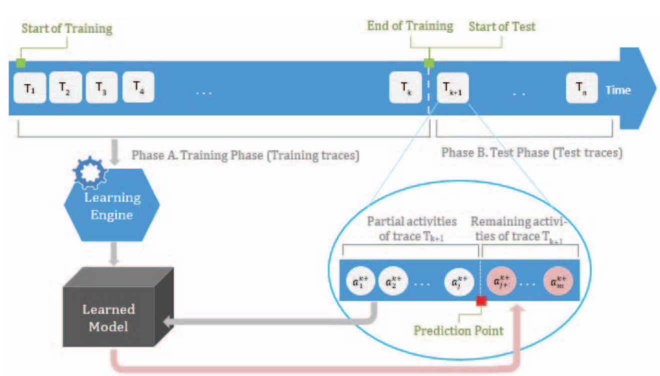

Fig. 1. Predictive business process.

2325-6362/20/$31.00 ©2020 IEEE
DOI 10.1109/EDOC49727.2020.00030

of the ongoing trace, we will have a partial trace of length j at the predicting point. Hence, the trained model's input will be the partial trace and the output will be its next event(s).

Many studies have recently addressed the prediction of the future behavior of a running process instance [10], [13], [14]. Evermann et al. [15] proposed a method to predict the next event using embedded dimensions of the LSTM network. It is one of the first studies that have used deep learning techniques in process monitoring. In another related work, Tax et al. [11] predicts timestamp of events, as well. They used one-hot encoding instead of embedded dimensions to encode events. Further on, Camargo et al. [16] proposed a method that brings together strengths of [11] and [15]. It is based on embedded dimensions and addresses both numerical and categorical attributes in an event log. Lin et al. [17] proposed another method that tries to predict all attributes of an event in addition to its activity name.

Recently, a new stream of work in predictive business process monitoring leveraged deep learning techniques to unlock the potential business value locked in process execution event logs. These works use Recurrent Neural Networks, such as LSTM and GRU. These works suffer from misinformation and accuracy in long sequences as they use the last hidden state (as the context vector) for the purpose of predicting the next event [12]. On the other hand, in operational processes, traces may be very long, which makes the above methods inappropriate for analyzing them. In addition, in predicting the next events in a running case, some of the previous events should be given a higher priority.

To address these shortcomings, in this paper, we present a novel approach inspired by the notion of attention mechanism, utilized in Natural Language Processing and, particularly, in Neural Machine Translation [18]. Our proposed approach uses all hidden states to accurately predict future behavior and the outcome of individual activities. The underlying idea is that the prediction of the next activity does not require the context vector to contain all the information related to the entire prefix. Accordingly, attention mechanisms could be used to improve the accuracy of predictions. The proposed method is evaluated from two points of view. First, we aim to answer the question of whether the attention mechanism improves the next event prediction. Besides, the proposed method is compared with the other methods.

The rest of this paper is organized as follows. Section II provides a short summary of the related work on the use of deep learning techniques in process mining and predictive process monitoring.We present the proposed approach in detail in Section III. Section IV outlines our evaluation plan and provides the details of the experimental results and, finally, Section V summarizes the contributions and findings and discusses suggestions for future work.

II. Related Work and Background

The first part of this section discusses the related work done in this field. Then, some deep learning techniques used in this paper are succinctly introduced.

A. Related Work

Many attempts have been made to predict the future behavior of an ongoing trace. The traditional approaches tried to make predictions using models such as algorithms Association Rule Mining (ARM) [19] and Hidden Markov Models (HMM) [20]. Although such approaches are appropriate for prediction of future behavior, they fail to perform well when traces are complex.

Recently, deep learning techniques have been used frequently to handle the problem. Their popularity is reflected in Verenich et al. [10], which provided a systematic literature review and an extremely comprehensive taxonomy of the methods of remaining time prediction of a running case. They reviewed publications from 2005 to 2017. They demonstrated that the method using LSTM outperformed the other methods on 13 out of 16 databases and made the lowest average error. Another survey of prediction methods, which included 55 publications, was made by [21]. Evermann et al. [15] utilized LSTM for next activity prediction. Their approach was derived from Natural Language Processing (NLP). Next activity prediction through the NLP-based method called for equating an event log with a document, a trace with a sentence, and an event with a word. Input sequences were encoded through word embedding, which saves space.

Tax et al. [11] created another LSTM-based prediction method. It is different from the method [13] in that this one predicts the timestamps of the next events and the remaining cycle time, as well. While Evermann et al. used embedded dimensions to encode events, Tax et al. fulfilled this purpose by one-hot encoding. As an advantage, Tax et al. enriched the feature vector with the features relating to the details of the event's occurrence time such as time of the day, the period between the current event and the previous one, and the temporal distance between the start of the case and the current point. However, the problem with one-hot encoding is that if the activities are large in number, it requires a large number of dimensions. They used three different LSTM architectures for prediction.

The method proposed by Camargo et al. [16] combines the advantages of [11], [15]. They encoded numerical attributes in addition to event types. The encoding took advantage of embedded dimensions, which saves space when there is a large number of event (activity) types. The method included a preprocessing phase in which the feature vectors were built. Then, the three architectures proposed in [11] were used for the predictions and the results were better than those obtained by [11]. Lin et al. [17] put forth a method that predicts all the attributes of an event in addition to the event type (activity). They suggested that it is required for the next event prediction to consider the relative importance of each attribute in a given event. The degree of importance was determined by a modulator. They also used an encoder-decoder architecture.

Clustering was presented, in [22], as a way of saving space when encoding events. The clustering made use of dependencies between attributes of events. Navarin et al. [23]

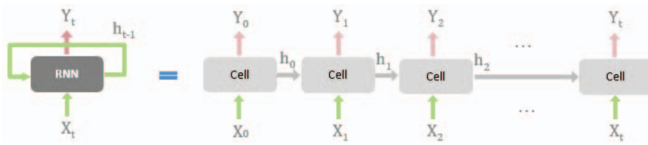

Fig. 2. An unrolled recurrent neural network.

focused on LSTM networks for data-aware remaining time prediction of business process instances, and addressed the prediction of the remaining time until the end of the case.

Note that all of the above studies try to enrich the feature vector or reduce the space needed for representing the feature vector. They ignore the fact that the prediction of the future behavior of an ongoing trace does not require paying equal attention to the input time steps. Rather, some of them must be given more and some less attention. There are other studies in the literature that used deep learning techniques: Dorgo et al. [24] utilized a seq2seq architecture to predict future events. A method called BINet was applied in [25]. It is a multi-variant neural network for real-time anomaly detection based on the next event prediction and next attribute prediction. Another study dealing with anomaly detection is [26], in which autoencoders were used for this purpose.

B. Background

It seems reasonable to precede the main discussion of our study with a succinct introduction to the relevant concepts.

1) Recurrent Neural Network: In a text, as an instance of sequential information, the words are interrelated and cannot be dealt with independently while learning a neural network. Therefore, when processing sequential information, traditional neural networks (feed-forward neural networks) may no longer be used. Recurrent Neural Networks (RNN) use loops to transfer information from one step to another [27]. This guarantees that the information of the previous time steps is kept and the interrelations between the words are captured. An RNN is shown in Figure 2.

The problem of vanishing and the exploding gradient is observed in RNNs. That is why they exhibit poor performance in long sequences. This gave rise to LSTM networks.

2) Long Short-Term Memory: Changes made to recurrent neural networks resulted in Long Short-Term Memory (LSTM) networks, which can remember previous information. The problem of vanishing gradient is resolved in LSTM. Learning the model is carried out via back-propagation. There are three gates in an LSTM network: Input gate, Forget gate, Output gate

3) Sequence to Sequence model: Introduced for the first time in 2014 by Sutskever et al. [28], a sequence to sequence model aims to map a fixed-length input with a fixed-length output where the length of the input and output may differ. Figure 3.

The model consists of 3 parts: encoder, context vector, and decoder. The encoder is a stack of several recurrent units (here is LSTM) where each accepts a single element of the input sequence, collects information for that element, and propagates it forward. The final state of the encoder is a context vector. This context vector aims to encapsulate the information for all input elements in order to help the decoder make accurate predictions. It acts as the initial hidden state of the decoder part of the model. The decoder is a stack of several recurrent units where each predicts an output y_t at a time step t. Each recurrent unit accepts a hidden state from the previous unit and produces and output as well as its own hidden state.

III. Proposed Approach

This section describes our method of predicting the next activity and the remaining sequence. As discussed in the previous section, the seq2seq model receives input sequences one at a time and finally encodes all of them in a fixed-length vector called context vector. The context vector then inputs to the decoder. The problem with this model is that the fixed length of the vector prevents the memorization of information in long sequences.

Consequently, the information carried by the earlier parts of the sequence is forgotten. The attention mechanism, which was introduced by Bahdanau et al. [18], is a solution to this problem. Since traces are often long in operational processes, this mechanism can lead to more accurate predictions of the behavior of the ongoing process.

A. Next Activity Prediction

This paper takes advantage of the notion of attention mechanism in the next activity prediction. This means that it is not only the final hidden state that is involved in the construction of the context vector, but each single hidden state relating to a time step is taken into consideration. The use of attention mechanism for next activity prediction involves attaching dissimilar degrees of importance to the prefix elements. In other words, since all the hidden states participate in the generation of the context vector, each individual hidden state has to be given its own particular coefficient. A coefficient represents how much attention the related input must receive. The architecture of our method for the next activity prediction is illustrated in Figure 5.

As shown in Figure 4, all input activities are passed through an embedding layer. Since neural networks operate on real-valued data, every input must be encoded before it goes to be processed in the neural network. A method for encoding

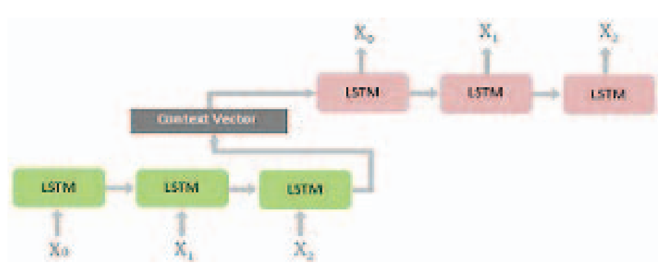

Fig. 3. Encoder-decoder sequence to sequence model [28]

.

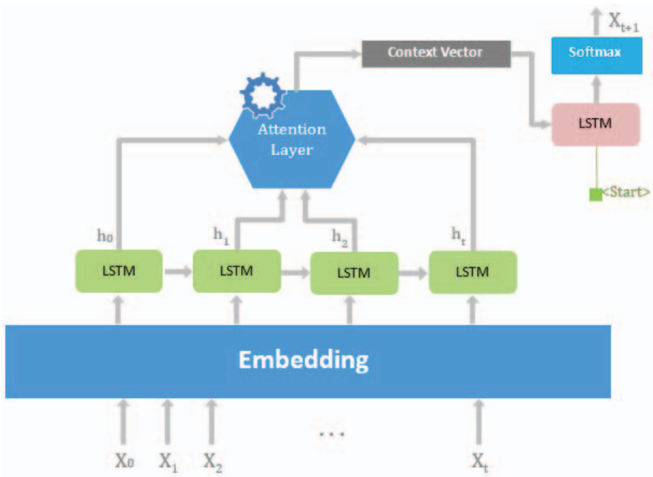

Fig. 4. Next event prediction by attention mechanism.

inputs is one-hot encoding, whose dimensions equal the size of the vocabulary, i.e. the number of unique activity names in the case of our study. This method occupies a large space. Therefore, word embedding is used to encode the inputs. This way, we will have a $v \times m$ matrix, where v denotes the number of activity names and m the size of the embedding vector. The values in each cell, which are of numeric type, are trainable. Having gone through the embedding layer, the input sequences, i.e. partial traces in this study, are sent to the encoder one at a time and the hidden states of the steps are created. In the decoding phase, the decoder is not restricted to only the final hidden state of the encoder, but rather all hidden states of the encoder are given to an attention layer. It is there that the decision is made about the coefficients to be assigned to the individual hidden states and, also, the context vector is constructed for the decoder. It should be added that the attention layer is a feed-forward neural network that is trained together with the encoder and decoder in the training phase. The importance of each of the hidden states is calculated as follows:

$$c = \sum_{i=1}^{T_x} a_i h_i \tag{1}$$

$$a_i = softmax(e_i) \tag{2}$$

$$e_i = V_a^T tanh(W_a h_{final} + U_a h_i) \tag{3}$$

, where c represents the context vector and T_x, h_i and h_{final}, respectively, denote the number of activities, the hidden state at the i-th time step and the final hidden state. a_i is the coefficient, that is to say, the attention given to the i-th hidden state in the construction of the context vector for the decoder. Note that the sum of the attention coefficients across all the hidden states equals 1. W_a, U_a, V_a are weight matrices, which must be learned. In machine translation, there are as many context vectors as the number of output words. However, since this study aims to predict the next activity, only one context vector is constructed.

B. Remaining sequence prediction

It is desired to predict the future behavior of the ongoing trace. This requires knowing what the remaining sequence of activities in the current case will be. It makes it necessary to predict all the activities, and not only the immediately next activity but that will also happen by the end of the case. This is called the remaining sequence prediction in this paper. In remaining sequence prediction, the function predicting the next activity is called repeatedly until the token [EOC] is returned. In that case, the operation stops, and the generated suffix is returned as the output.

IV. Evaluation

The proposed method is experimentally evaluated from two points of view. The first one aims to find out whether the attention mechanism leads to improved prediction of next events. For this purpose, the proposed method is compared with the method of [15], which is similar in the approach but does not use the attention mechanism. The other evaluation seeks to compare the proposed method with state-of-the-art approaches. This goal is fulfilled by evaluating the proposed method on various datasets. The datasets used in this paper are introduced in Section A. Then, the implementation of the experiments, the metrics used for evaluation, and the evaluation results are discussed.

A. Datasets

The proposed method is evaluated on 4 different real-life datasets. The name and specifications of datasets, which come from different domains are listed in Table I:

B. Preprocessing

In the preprocessing stage, the datasets get prepared for being fed into the network. The above datasets are of the XES format and contain the full range of information on their event logs, part of which is not needed in this study. Since only the sequences of activities are needed for prediction of next activities, the other attributes are left out and only the activity names of the events are retained. The activities involved in a single case are placed in a single row represented by the caseID. Prediction of the next activity essentially requires partial traces. The partial traces gathered from the datasets are required to be of the minimum length of 5, which is the

TABLE I
DATASET SPECIFICATION

Dataset	*#Case*	*#activities*	*Min case length*	*Max case length*	*Avg case length*
Helpdesk	4580	14	3	15	2.9
BPIC12	13087	24	3	96	15.9
BPIC12W	9658	7	1	74	13.5
BPIC12O	5015	8	3	30	6.8

minimum length adopted by the other methods in the literature. This gives us the possibility of comparing our method with other methods. However, since the sequences in the Helpdesk dataset are usually short, the minimum length of a sequence is set to 2 for that dataset. If the length of a case is equal to or greater than the minimum required length, it is converted into several sequences. For instance, if the length of a case is 8, it is converted into three sequences of the lengths of 5, 6, 7, each of which has a particular target activity. To mark the end of each case, a [EOC] token is appended at the end of it. Each dataset is split into two training and testing sets. 70% of each dataset is used for training our model and 30% is allocated to the testing set.

C. Experimental Results

In this section, the results produced by the proposed method are assessed. As mentioned in the previous section, the proposed method deals with the prediction of both the next activity and the remaining sequence. The evaluation of the quality of the predictions, by means of two metrics, is discussed in the following subsections.

1) Next activity prediction: To evaluate the performance of the next activity prediction, the accuracy metric is used. After the model is trained on the training data, 30% of the data is used for testing. It means that a partial trace is input to the model and the next activity predicted by the model is compared with the ground-truth. The number of correct predictions is divided by the total number of sequences in the testing set. In Table II, the proposed method is compared with the other methods in terms of how well they carry out the next activity prediction in the four datasets.

Table II suggests that the proposed method outperforms the methods presented by [11], [15] and [16] in the case of the datasets BPIC12 and Helpdesk. Nevertheless, it underperforms [17] for the same datasets. It should be noted that, when encoding events, the other methods, except [15], enrich the feature vector with a variety of attributes, including resource, timestamp, etc. However, in the next activity prediction, this paper, as well as [15], relies only on activity name and excludes the other attributes of events. In order to evaluate the effect of attention mechanism, the proposed method must be compared with the baseline method of [15], which includes only activity name. As demonstrated, for all datasets, attention mechanism, which is used in the proposed method, leads to much better results in comparison with baseline. Even considering the exclusion of the attributes other than activity name, the proposed method still outperforms the other methods in the case of most datasets.

2) Remaining sequence prediction.: The performance of the remaining sequence prediction is assessed by the Demerau-Levinstain (DL) algorithm [29]. DL serves to measure the distance between two sequences. The idea is to apply as many editions as needed to the two sequences so that they become completely identical. The distance between the two sequences is equated with how many editions are needed for this purpose. For each of the four editions of insertion, deletion, substitution,

TABLE II
THE ACCURACY OF NEXT ACTIVITY PREDICTION

Dataset	*Evermann et al. [15]*	*Tax et al. [11]*	*Lin et al. [17]*	*Camargo et al. [16]*	*Proposed approach*
Helpdesk	0.798	0.712	0.916	0.789	0.833
BPIC12	0.788	-	0.974	0.786	0.816
BPIC12W	0.658	0.760	-	0.778	0.723
BPIC12O	0.836	-	-	-	0.839

TABLE III
THE AVERAGE SIMILARITY BETWEEN THE PREDICTED REMAINING SEQUENCES AND THE REAL SEQUENCES OBSERVED IN THE DATASETS

Dataset	*Evermann et al. [15]*	*Tax et al. [11]*	*Lin et al. [17]*	*Camargo et al. [16]*	*Proposed approach*
Helpdesk	0.742	0.767	0.874	0.917	0.965
BPIC12	0.110	-	0.281	0.632	0.703
BPIC12w	0.297	0.353	-	0.525	0.446

and transposition, penalty points are considered. Scaling the final calculated point by the maximum sequence length yields the quantified distance between the two sequences, i.e. the predicted sequence and the real-life sequence in the dataset. Subtracting this from unity yields the degree of similarity between the two sequences. The calculations are carried out as follows.

$$Sim(s_1, s_2) = 1 - \frac{DL(s_1, s_2)}{max(len(s_1), len(s_2))} \quad (4)$$

Once a sequence is predicted through the proposed method, it is compared, in pairs, with any real-life sequences that start with the partial trace and the pairwise similarities are determined. The highest degree of similarity is taken into account for the particular partial trace in question. The mean value of the similarities over the entire range of the predicted sequences provides the average similarity between the predicted remaining sequence and the ground-truth.

As shown in Table III, the proposed approach outperforms all of the state-of-the-art methods. However, it underperforms the approach by Camargo et al. [16] in the case of BPIC12W.

V. CONCLUSION AND FUTURE WORK

This study improved LSTM with attention mechanisms to predict the future behavior of a running case. The idea was derived from Natural Language Processing, and in particular, Neural Machine Translation. The problem with LSTM is that it fails to remember the past in long sequences and misses information. Since sequences of operational processes are usually long, LSTM leads to low accuracy of prediction. The attention mechanism is a solution to this problem. The rationale behind the attention mechanism is that all-time steps are not equally significant for prediction of the future behavior of a partial trace and some of them must be given more attention. Thus, for instance, sometimes only a couple of activities located at the end of a partial trace are needed while, on some occasions, an activity appearing in the middle of a partial trace receives attention.

The evaluation of the proposed method revealed that, due to the event logs containing long sequences, the attention mechanism leads to more accurate predictions of the future behavior of a running case. Prediction of the remaining sequence, in particular, is an area in which the proposed method outperforms, by a margin, the state-of-the-art.

This study, in contrast to the methods forming the basis of comparison, considers only the activity name of an event and excludes the other attributes for prediction of the future behavior of a running case. Nevertheless, taking such valuable information as the resource, timestamp, and even the location of an event can improve prediction. In addition, including a timestamp in the feature vector makes it possible to predict the next timestamp and the remaining cycle time. Thus, we will attempt to create a new method for encoding all attributes of an event and applying attention mechanisms to the prediction of any attribute and not only the activity name of an event. Also, since the proposed method adds an attention layer to LSTM, training time gets longer. Therefore, it is necessary to carry out an evaluation from the viewpoint of time, as well. Finally, we will evaluate the proposed method for more datasets.

Acknowledgement. We acknowledge the AI-enabled Processes (AIP[1]) Research Centre for funding this research.

References

[1] W. Van Der Aalst, "Data science in action," in *Process mining*, pp. 3–23, Springer, 2016.

[2] M. Dumas, M. La Rosa, J. Mendling, and H. A. Reijers, "Introduction to business process management," in *Fundamentals of business process management*, pp. 1–31, Springer, 2013.

[3] S.-M.-R. Beheshti, B. Benatallah, S. Sakr, D. Grigori, H. R. Motahari-Nezhad, M. C. Barukh, A. Gater, S. H. Ryu, *et al.*, *Process analytics: concepts and techniques for querying and analyzing process data*. Springer, 2016.

[4] A. Beheshti, B. Benatallah, and H. R. Motahari-Nezhad, "Processatlas: A scalable and extensible platform for business process analytics," *Softw. Pract. Exp.*, vol. 48, no. 4, pp. 842–866, 2018.

[5] A. Beheshti, F. Schiliro, S. Ghodratnama, F. Amouzgar, B. Benatallah, J. Yang, Q. Z. Sheng, F. Casati, and H. R. Motahari-Nezhad, "iprocess: Enabling iot platforms in data-driven knowledge-intensive processes," in *Business Process Management Forum - BPM Forum 2018, Sydney, NSW, Australia, September 9-14, 2018, Proceedings*, vol. 329 of *Lecture Notes in Business Information Processing*, pp. 108–126, Springer, 2018.

[6] S. Beheshti, B. Benatallah, H. R. M. Nezhad, and S. Sakr, "A query language for analyzing business processes execution," in *Business Process Management - 9th International Conference, BPM 2011, Clermont-Ferrand, France, August 30 - September 2, 2011. Proceedings* (S. Rinderle-Ma, F. Toumani, and K. Wolf, eds.), vol. 6896 of *Lecture Notes in Computer Science*, pp. 281–297, Springer, 2011.

[7] W. M. van der Aalst, M. Bichler, and A. Heinzl, "Robotic process automation," 2018.

[8] F. Amouzgar, A. Beheshti, S. Ghodratnama, B. Benatallah, J. Yang, and Q. Z. Sheng, "isheets: A spreadsheet-based machine learning development platform for data-driven process analytics," in *Service-Oriented Computing - ICSOC 2018 Workshops - ADMS, ASOCA, ISYyCC, CloTS, DDBS, and NLS4IoT, Hangzhou, China, November 12-15, 2018, Revised Selected Papers*, vol. 11434 of *Lecture Notes in Computer Science*, pp. 453–457, Springer, 2018.

[9] F. Schiliro, A. Beheshti, S. Ghodratnama, F. Amouzgar, B. Benatallah, J. Yang, Q. Z. Sheng, F. Casati, and H. R. Motahari-Nezhad, "icop: Iot-enabled policing processes," in *Service-Oriented Computing - ICSOC 2018 Workshops - ADMS, ASOCA, ISYyCC, CloTS, DDBS, and NLS4IoT, Hangzhou, China, November 12-15, 2018, Revised Selected Papers*, vol. 11434 of *Lecture Notes in Computer Science*, pp. 447–452, Springer, 2018.

[10] 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.

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

[12] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[13] 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.

[14] F. Schiliro, A. Beheshti, and N. Moustafa, "A novel cognitive computing technique using convolutional networks for automating the criminal investigation process in policing," in *Proceedings of SAI Intelligent Systems Conference*, pp. 528–539, Springer, 2020.

[15] J. Evermann, J.-R. Rehse, and P. Fettke, "Predicting process behaviour using deep learning," *Decision Support Systems*, vol. 100, pp. 129–140, 2017.

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

[17] L. Lin, L. Wen, and J. Wang, "Mm-pred: a deep predictive model for multi-attribute event sequence," in *Proceedings of the 2019 SIAM International Conference on Data Mining*, pp. 118–126, SIAM, 2019.

[18] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[19] C. Rudin, B. Letham, A. Salleb-Aouissi, E. Kogan, and D. Madigan, "Sequential event prediction with association rules," in *Proceedings of the 24th annual conference on learning theory*, pp. 615–634, 2011.

[20] M. Le, B. Gabrys, and D. Nauck, "A hybrid model for business process event and outcome prediction," *Expert Systems*, vol. 34, no. 5, p. e12079, 2017.

[21] C. Di Francescomarino, C. Ghidini, F. M. Maggi, and F. Milani, "Predictive process monitoring methods: Which one suits me best?," in *International Conference on Business Process Management*, pp. 462–479, Springer, 2018.

[22] M. Hinkka, T. Lehto, and K. Heljanko, "Exploiting event log event attributes in rnn based prediction," in *Data-Driven Process Discovery and Analysis*, pp. 67–85, Springer, 2018.

[23] N. Navarin, B. Vincenzi, M. Polato, and A. Sperduti, "Lstm networks for data-aware remaining time prediction of business process instances," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–7, IEEE, 2017.

[24] G. Dorgo, P. Pigler, M. Haragovics, and J. Abonyi, "Learning operation strategies from alarm management systems by temporal pattern mining and deep learning," in *Computer Aided Chemical Engineering*, vol. 43, pp. 1003–1008, Elsevier, 2018.

[25] T. Nolle, A. Seeliger, and M. Mühlhäuser, "Binet: Multivariate business process anomaly detection using deep learning," in *International Conference on Business Process Management*, pp. 271–287, Springer, 2018.

[26] T. Nolle, S. Luettgen, A. Seeliger, and M. Mühlhäuser, "Analyzing business process anomalies using autoencoders," *Machine Learning*, vol. 107, no. 11, pp. 1875–1893, 2018.

[27] F. J. Pineda, "Generalization of back-propagation to recurrent neural networks," *Physical review letters*, vol. 59, no. 19, p. 2229, 1987.

[28] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, pp. 3104–3112, 2014.

[29] F. J. Damerau, "A technique for computer detection and correction of spelling errors," *Communications of the ACM*, vol. 7, no. 3, pp. 171–176, 1964.

[1]https://aip-research-center.github.io/