Object-centric predictive process monitoring

Wissam Gherissi, Joyce El Haddad, Daniela Grigori

Object-centric predictive process monitoring

Wissam Gherissi[1][0000−0002−2234−2963], Joyce El Haddad[1][0000−0002−2709−2430], and Daniela Grigori[1][0000−0003−1741−8676]

Université Paris-Dauphine, Université PSL, CNRS, LAMSADE, 75016 PARIS, FRANCE
{wissam.gherissi,joyce.elhaddad,daniela.grigori}@dauphine.psl.eu

Abstract. Predictive process monitoring approaches aim to make predictions about the future behavior for running instances of business processes, such as next activity or remaining time. Most of these approaches use single object type event logs as if the business process is operating in isolation. Whereas, in an organization, several instances of different processes related to a set of objects can be executed at the same time and may interact with each other. This paper investigate the use of object-centric event logs as they offer information about events and their related objects, allowing access to a global view about the running processes in an organization. We propose an object-centric predictive approach considering interactions between different object types. The proposed approach is evaluated on a publicly available object-centric log. The analysis of the results shows that using additional features (i.e., several object types' information) can generally help increase prediction performances.

Keywords: Predictive Monitoring · Object-centric Event Log · Process Mining.

1 Introduction

Process Mining has seen many developments over the last years with the advances of artificial intelligence domain, especially in machine and deep learning. Researchers have adapted innovative machine learning models and architectures to apply them on the different process mining techniques like process discovery, conformance checking and process enhancement. Process mining takes as input an event log recording events related to the execution of a process. The proposed techniques until recently worked on single case identifier event logs. This type of event logs is limited to a single perspective i.e. single type of object of a process whereas real processes can have multiple perspectives and incorporate multiple objects. These event logs have limited the use of process mining to studying a precise aspect of a process rather than all interactions between objects and processes, and also posed other problems like convergence and divergence [1] due to differences between the event logs and the reality of processes. Object-centric event logs (OCEL) are rather different than classic event logs with closer representation of real processes. OCELs store information about the process on

multiple aspects in one event log with the use of object types. OCELs allow access to the process covering all the related instances. For example in the case of order management process, an OCEL will contain information about instances ranging from placing the order, products management until the package delivery. Recently, there have been general interest in the use of OCELs for process mining especially in process discovery using Petri nets [4] and performance analysis [9]. But, to the best of our knowledge, there is still no approach tailored for OCEL for predictive process monitoring, especially for behavior prediction. When we try to use OCEL for prediction tasks, the first problem we face is identifying the trace ID. In OCEL, an event is not associated with a single case ID, as in classical logs. While classical logs can be obtained by flattening the log, using these logs in isolation may not take into account the interaction between objects.

In this paper, we analyze the specific challenges that OCEL poses for predictive monitoring and we investigate whether taking into account the interaction between objects may improve the prediction. We propose a first approach for predictive monitoring, more precisely next activity and time prediction (next event and remaining time).

We adapt the LSTM architecture inspired from [17], that have been proven to have very good results on traditional event logs. We extend this architecture to predict instance remaining time and obtain better results than predicting activities iteratively until the end of the sequence. We evaluate our model and approach on a publicly available OCEL [1] and we made available the code on Github [2].

The remainder of the paper is as follows. Section 2 discusses related work on object-centric event logs and predictive process monitoring. Section 3 introduces the basic notions and formal definitions for event logs. Section 4 presents the proposed LSTM model and the approach from data preprocessing to prediction tasks. Section 5 presents the experimental results and Section 6 concludes the paper.

2 Related Work

To the best of our knowledge, there are no dedicated works that examine both predictive process monitoring and object-centric event logs. To address this gap, we discuss first some works on object-centric event logs and then some works on predictive process monitoring.

Object-centric process mining. Process mining [3] is a set of techniques that can be applied on event logs such as process discovery to detect patterns and model processes. There are also techniques like conformance checking and process enhancement with the objective of improving the process performance and making sure that the process is conforming to established norms.

[1] `ocel-standard.org/1.0/running-example.jsonocel.zip`
[2] `https://github.com/wissam-gherissi/PPM-OC`

Recently, researchers have been studying a new format of event logs called object-centric event logs [10] as an alternative to traditional single case event logs to overcome problems like "convergence" and "divergence" mentioned in [1] and present new insights on processes.

In order to represent multiple processes in the same event log, [2] propose the idea of federated process mining utilizing event data from different sources of information in the same organization or across organizations. Other papers working on this type of event logs evolve around the technique of process discovery by discovering the process Petri net in [4] or the behavioral constraints model in [13], hence discovering the different relations between the activities and multiple object types. Furthermore, the notions of precision and fitness as metrics to evaluate process discovery have been adapted for the case of object-centric event logs in [5].

Predictive process monitoring. In predictive process monitoring papers, different approaches and models were applied on traditional event logs for a variety of prediction tasks. For next activity and time remaining predictions, models were proposed such as recurrent neural networks (RNNs) [11] [12], convolutional neural networks (CNNs) [15] [8], attention mechanisms [16] and transformers [6].

LSTMs were studied in [17], [7] and [14] as a solution for sequence prediction given the long memory capacity of this architecture. Precisely in [17], the model is constructed using LSTM layers for multiple prediction tasks (next activity, next event time, suffix prediction, time remaining prediction) using one model with a single training step for different tasks rather than training multiple models divided over the tasks. We choose this architecture for its performances on classic logs. We adapt it for OCEL, by introducing changes in the data preprocessing part and to the model construction, adding a LSTM layer for the remaining time prediction task.

All of these previous works presented in this part have focused on implementing prediction models and studying patterns for traditional event logs rather than object-centric event logs.

Finally, the closest paper to our approach is working on the combination of object-centric event logs and predictive process analytics in [9]. This work, although used object-centric event logs, is focused on predictive analysis and performance indicators using gradient boosting. In this paper, we use neural networks and LSTMs for predicting process behavior, precisely next activity, next event time and remaining time using object-centric event logs.

3 Preliminaries

In this section, we introduce the basic notions for object-centric event logs and LSTMs that will be used in this paper.

3.1 Object centric event logs

These notions are based on the definitions of [10].

Definition 1. *(Universes) The used universes are:* $\mathbb{U}_E$ *the universe of events,* $\mathbb{U}_{act}$ *the universe of activities,* $\mathbb{U}_{att}$ *the universe of attribute names,* $\mathbb{U}_{val}$ *the universe of attribute values,* $\mathbb{U}_{typ}$ *the universe of attribute types,* $\mathbb{U}_o$ *the universe of object identifiers,* $\mathbb{U}_{ot}$ *is the universe of object types,* $\mathbb{U}_{act}$ *the universe of activities,* $\mathbb{U}_{time}$ *the universe of timestamps.*

Definition 2. *(OCEL) An object-centric event log is defined in [10] as a tuple* $L = (E, AN, AV, AT, OT, O, \pi_{typ}, \pi_{act}, \pi_{time}, \pi_{vmap}, \pi_{omap}, \pi_{otyp}, \pi_{ovmap}, <)$ *where,*

- $E \subseteq \mathbb{U}_E$ *is the set of events.* $AN \subseteq \mathbb{U}_{att}$ *is the set of attribute names.* $AV \subseteq \mathbb{U}_{val}$ *is the set of attribute values.* $AT \subseteq \mathbb{U}_{typ}$ *is the set of attribute types.* $O \subseteq \mathbb{U}_o$ *is the set of object identifiers.* $OT \subseteq \mathbb{U}_{ot}$ *is the set of object types.*
- $\pi_{typ} : AN \cup AV \rightarrow AT$ *is the function associating each attribute name or value to the corresponding attribute type.* $\pi_{act} : E \rightarrow \mathbb{U}_{act}$ *is the function associating each event with its activity.* $\pi_{time} : E \rightarrow \mathbb{U}_{time}$ *is the function associating each event with a timestamp.*
- $\pi_{vmap} : E \rightarrow (AN \not\rightarrow AV)$ *such that:*

$$\pi_{typ}(n) = \pi_{typ}(\pi_{vmap}(e)(n)) \quad \forall e \in E, \forall n \in \mathrm{dom}(\pi_{vmap}(e))$$

  is the function associating an event identifier to its attribute value assignments with the condition that the attribute name and value have the same attribute type.
- $\pi_{omap} : E \rightarrow \mathcal{P}(O)$ *is the function associating an event identifier to a set of related object identifiers. In the case of additional objects identifiers used in the definition 4, we use the notion of* π_{rmap}
- $\pi_{otyp} \in O \rightarrow OT$ *is the function assigning precisely one object type for each object identifier.*
- $\pi_{ovmap} : O \rightarrow (AN \not\rightarrow AV)$ *such that*

$$\pi_{typ}(n) = \pi_{typ}(\pi_{ovmap}(o)(n)) \quad \forall o \in E, \forall n \in \mathrm{dom}(\pi_{vmap}(o))$$

  is the function associating an object to its attribute value assignments. Each object is related to attributes with attribute name and value such that the name and the value must have the same attribute type.
- $<$ *is a partial order based on the timestamps of events such that*

$$e_1 < e_2 \iff \pi_{time}(e_1) < \pi_{time}(e_2)$$

An example of an OCEL is shown in Table 1 for the process of ordering products for deliver. As it can be seen, the order have many related objects such as the ordered items and the packages in which the delivery is happening.

Definition 3. *(Single object type event log) Inspired from the Object type Projection definition in [1], a single object type event log (an example is shown in Table 2) is defined as follows:*

Event identifier	Activity name	Timestamp	Objects			Attributes	
			Order	Item	Package	Price	Weight
...	...	...	...	...	...	...	...
70	place order	2019 − 05 − 22 10 : 33 : 18	{990018}	{880074, 880075, ...}	∅	823.98	1.97
71	confirm order	2019 − 05 − 22 10 : 34 : 00	{990008}	{880023, 880024, ...}	∅	1428.97	3.75
...	...	...	...	...	...	...	...
14230	pick item	2019 − 12 − 31 17 : 17 : 59	{991296}	{885262}	∅	29.99	0.38
14231	item out of stock	2019 − 12 − 31 17 : 20 : 10	{991291}	{885233}	∅	79.99	0.483
...	...	...	...	...	...	...	...
17900	create package	2020 − 02 − 24 13 : 45 : 30	{991572, 991393, ...}	{886439, 886438, ...}	{661050}	1424.97	2.852
17901	send package	2019 − 12 − 31 13 : 52 : 48	{991583, 991604, ...}	{886559, 886479, ...}	{661048}	2749.97	3.01
...	...	...	...	...	...	...	...

Table 1. Example of object centric event log describing order management process

$(E^{ot}, \leq^{ot})$ *is a projection of the events of an object-centric event log on a single object type, where for a specific* $ot \in \mathbb{U}_{ot}$

$e \in E^{ot}$ *such as* $e = (ei, act, time, omap, vmap), E^{ot} = \{e \in E \mid \pi_{omap}(e)(ot) \neq \emptyset\}$ *and*

$$\preceq_E^{ot} = \{(e_1, e_2) \in E^{ot} \times E^{ot} \mid e_1 \preceq_E e_2 \wedge \pi_{omap}(e_1)(ot) \cap \pi_{omap}(e_2)(ot) \neq \emptyset\}$$

Case identifier	Activity name	Timestamp
...	...	...
990018	place order	2019 − 05 − 22 10 : 33 : 18
990008	confirm order	2019 − 05 − 22 10 : 34 : 00
...	...	...
991296	pick item	2019 − 12 − 31 17 : 17 : 59
991291	item out of stock	2019 − 12 − 31 17 : 20 : 10
...	...	...
991572	create package	2020 − 02 − 24 13 : 45 : 30
991583	send package	2019 − 12 − 31 13 : 52 : 48
...	...	...

Table 2. Extracting single object type event log on order object type from Table 1

4 Predictive process monitoring for object-centric logs

As we have seen in the previous section, multiple object types might interact and influence each other in a process, resulting in multiple case notions. Our goal is to handle these case notions in predictive monitoring. With an object-centric log, user may be interested in :

- P1: making predictions about a given object (next activity for a given item, delivery date of a given package)
- P2: making predictions about the global process (in an order management, what is the next activity concerning the related items, packages, etc)

For the first type of prediction (P1), it is possible to use the single object type to make predictions for a given object. However, in some cases, user could be interested in defining a custom object life cycle. Thereby, we define a filtered log notion.

For the second type of prediction (P2) for a global process, using the single object type log may not be sufficient, as the interactions between objects are not taken into account. For this reason, we propose the enriched log notion.

Consider the process of ordering products for delivery (whose log is presented in Table 1), where an order is placed first by the customer for the purchase and delivery of products. In order to consider the instance of an order as completed, all the other related objects (items, packages) have to have their instances completed. In this case, the orders is the global process. It is clear that predicting the remaining time until the completion of the order instance should take into account the interactions between objects.

In the following, we define the notions of enriched and filtered log useful for object-centric predictive monitoring.

First, we define a new type of event log where each event, in addition to the single object type projection, is enriched with a set of the other related objects identifiers.

Definition 4. *(Enriched single object type event log) An enriched single object type event log* $(E^{ot}_{enr}, \preceq^{ot}_{enr})$ *(an example is shown in Table 3) is a projection of an object-centric event log on a single object type* ***ot*** *with the addition of an event attribute* ***rmap*** *describing the set of related object identifiers with types* ***ro*** *included in* ***enr*** *:*

$$e \in E^{ot}_{enr} \text{ such as } e = (ei, act, time, rmap, vmap)$$

$$E^{ot}_{enr} = \{e \in E \mid \pi_{omap}(e)(ot) = ei, \forall ro \in enr, \pi_{rmap}(e)(ro) \neq \emptyset\} \text{ and}$$

$$\preceq^{ot}_{enr} = \{(e_1, e_2) \in E^{ot} \times E^{ot} \mid e_1 \preceq_E e_2 \wedge \pi_{omap}(e_1)(ot) \cap \pi_{omap}(e_2)(ot) \neq \emptyset\}$$

Event identifier	Activity name	Timestamp	Related Objects	
			Item	Package
...	...	...	...	...
990018	place order	2019 − 05 − 22 10 : 33 : 18	{880074, 880075, ...}	∅
990008	confirm order	2019 − 05 − 22 10 : 34 : 00	{880048, 880049, ...}	∅
...	...	...	...	...
991296	pick item	2019 − 12 − 31 17 : 17 : 59	{885262}	∅
991291	item out of stock	2019 − 12 − 31 17 : 20 : 10	{885233}	∅
...	...	...	...	...
991572	create package	2020 − 02 − 24 13 : 45 : 30	{886439, 886438, ...}	{661050}
991583	send package	2019 − 12 − 31 13 : 52 : 48	{886559, 886479, ...}	{661048}
...	...	...	...	...

Table 3. Example of enriched single object type event log on order object type from Table 1

The simple object log is enriched with information about the related objects. Thus object interaction is extracted, allowing to take into account that a specific object type process evolution is affected by the progress made by other related objects processes.

Definition 5. *(Filtered simple object event log) A filtered simple object event log is a simple object event log where only events concerning activities of interest for the user are kept.*

This allows an analyst to define the object cycle which is of interest for him for the predictive monitoring, by eliminating activities which do not influence or are not relevant for the object life cycle. An example of a filter is shown in Table 4.

4.1 Pipeline for predictive process monitoring

The workflow of our approach is detailed below. It consists of three main steps: event log preprocessing, feature engineering, and prediction model construction.

Event log preprocessing. The first step of the approach is extracting the types of logs we described above by first flattening the OCEL over specific object types. Flattening an object-centric event log is transforming it into a single object type event log as per Definition 3. In the case of an event having multiple objects of the same type, that event is extracted in each separate "flattened" event log maintaining pertinent traces for all objects (this corresponds to the convergence phenomenon which, contrary to process modeling, is not a problem for process analysis).

As explained above, in some cases, an additional filter on the event log must be applied in order to specify the object types of interest for each activity resulting in a filtered log as per definition 5. An example is shown for further details in the Section 5.

Starting from the flattened log for a given object, the enriched single object type event log (Definition 4) will be built by adding an additional attribute, namely the set of objects related to the given object. As for example, in the order management example, an event can refer to multiple object types used like orders, items and packages.

Feature engineering. In this step, the goal is to prepare the input data for the prediction model. First, in the case of deep learning models, input data has to be of fixed shape. The input will be a multi-dimensional matrix containing features for each activity of each case. After calculating the maximum trace length (i.e., the number activities describing the execution of one case), its dimensions are defined as $number\_of\_cases \times max\_trace\_length \times number\_of\_features$.

The features set for the input matrix contain both categorical and numerical attributes. Categorical features describe the activity taking place during each event. Numerical features describe temporal properties of the event (time duration between the event and the start of the sequence, time since the last event, time of the day (since midnight) and the day of the week). These features also describe the related objects set in the enriched event log by calculating their number (in our case, we calculate the number of related items).

Prediction model construction. The prediction tasks are multi-class classification for next activity prediction and regression for next event time prediction and instance time remaining. In the first task, for each event and based on a fixed size prefix, the model predicts the probability for each activity being the

next event and the activity with the highest probability will be selected as the model's prediction for next activity. For the second task, the model predicts the duration after which the next activity will need to be executed. For the remaining time prediction, the model predicts the duration separating the present time step and the end of the instance. In [17], remaining time can be predicted using the same model by using a loop to predict the next event time until the model predicts the end of the sequence and the remaining time will be the sum of predictions. In this paper, we applied a modification to the model architecture by adding an LSTM layer for predicting the instance remaining time directly instead of looping over the same model for multiple prediction, thus decreasing the error rates.

Our implemented model is inspired by the architecture used in [17]. It is composed of four LSTM layers of size 100, the first layer is shared for all tasks taking on the data matrix as input, the other three layers are divided into the prediction tasks taking in as input the output of the shared layer.

The next activity layer feeds its output to a fully connected layer with softmax activation function [3], the output size is equal to the number of unique activities plus one class that will characterize the end of a trace, thus calculating the probabilities of belonging to each class. The time prediction layers feed the output to a fully connected layer with output of size 1 to predict the time remaining until the next activity or until the end of the sequence.

All performances recorded below are the mean of the performances on all prefixes length ranging from 2 to maximum trace length - 1. For next activity prediction, we use the accuracy metric. For time prediction tasks, we evaluate the model's performance using MAE (Mean Absolute Error) in days.

The implemented model is further summarized in the figure 1:

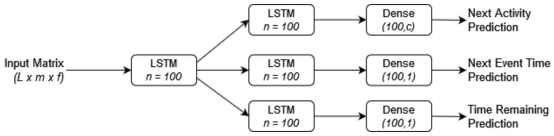

Fig. 1. Model architecture: *L: number of sentences, m: maximum trace length, f: number of features, n: number of neuros, c: number of unique*

5 Experimental Results

We implemented the approach presented in previous section in Python and we made the code publicly available[4] for reproducible results.

[3] Softmax function: $\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad for\ i = 1, 2, \ldots, K$

[4] `https://github.com/wissam-gherissi/PPM-OC`

The results presented were implemented on the following configuration: HP Elitebook 830 G8, processor: Intel Core i7-1185G7 @3.00GHz with 64GB RAM.

Data. We applied the approach propposed above on an order management object-centric event log [5]. The event log is composed of 22367 events and 5 objects types: "customers", "items", "orders", "packages" and "products". There are 17 different customers, 8159 items, 2000 orders, 1325 packages and 20 products.

In the enrichment step, we add a numerical feature describing the number of related items for each event. For example, for an event with the activity "place order", we add the number of items ordered during that event. For the package object type, we have the possibility of choosing orders or items or both as related objects. We limit the experiments on the items as related objects for this paper.

We focus on three object types (order, item and package), whose relationships are presented in figure 2. We expect that adding the other types (customer and product) would no have significant impact on predictions, given the cardinality of their relationship (one customer per order and one product per item). Furthermore, given these relationships between the items and the other object types (one order/package per item), we limit the enrichment step to the order and package object types. This results in "NA"(Not Applicable) values observed in Table 5 for the enriched event log on item level.

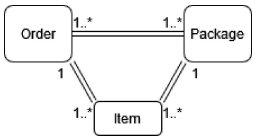

Fig. 2. Object types relationships overview

Prediction results. We present the prediction results of our approach for the different types of tasks we proposed. First, we suppose that the analyst is interested in making predictions for custom object life-cycle and then we present results on the complete object-life cycle and on the global process. We flatten the event log on the order, item and package object types in addition to a custom filter on the activity-object type combination presented in the table 4. The prediction results are presented in Table 5.A.

Table 5.B presents prediction results for complete event logs flattened on object types: order and item (without any filter on activities). For the packages, there are only four related activities; all these activities have been selected in the filter, thus the filtered log and the complete log are the same.

In Table 5.A, we observe satisfying results for the accuracy for next activity prediction with over 87% for order and item object types and over 75% for

[5] http://ocel-standard.org/1.0/running-example.jsonocel.zip

Activity	Object type
Place order	Order, item
Confirm order	Order, item
Payment reminder	Order
Pay order	Order
Item out of stock	Item
Reorder item	Item
Pick item	Item
Create package	Item, Package
Send package	Item, Package
Failed delivery	Item, Package
Package delivered	Item, Package

Table 4. Activity-Object type Filter

package with improvement for the case of enriched event log. For time prediction, we have order object type sequences with an average trace length of 20 days, we observe less than 7 days for next event time and 9 days for remaining time. For package object type, average trace length is 1.55 days. we have MAEs of 0.61 and 0.28 for next event time and remaining time respectively. For the items, we have an average trace length of 15.5 days. Error rates are 1.13 and 1.62 for next event time and remaining time.

Object type	Prediction Task	Single EL	Enriched EL
Order	Next Activity Accuracy	0.87	**0.88**
Order	Next Event Time MAE (days)	**6.70**	6.78
Order	Remaining Time MAE (days)	12.21/**8.70**	11.66/11.71
Package	Next Event Accuracy	0.75	**0.77**
Package	Next Event Time MAE (days)	0.61	**0.59**
Package	Remaining Time MAE (days)	0.85/0.29	0.82/**0.28**
Item	Next Event Accuracy	0.87	NA
Item	Next Event Time MAE (days)	1.13	NA
Item	Remaining Time MAE (days)	6.04/**1.62**	NA

Object type	Prediction Task	Single EL	Enriched EL
Order	Next Activity Accuracy	0.60	**0.63**
Order	Next Event Time MAE (days)	1.01	**0.95**
Order	Remaining Time MAE (days)	76/**8.2**	81/14
Item	Next Activity Accuracy	0.78	NA
Item	Next Event Time MAE (days)	1.62	NA
Item	Remaining Time MAE (days)	38/**5.62**	NA

Table 5. (A) Prediction results for the filtered event log and enriched flattened on order, item and package object types (left), (B) Prediction results for the complete event log and enriched flattened on order and item object types (right)

In Table 5.B, in next activity prediction, we observe accuracy over 60% for order and 78% for item. We notice improvement in terms of accuracy for the enriched event log for order object type. In next event time and remaining time predictions, we observe error rates of 0.95 and 8.2 respectively for order object type. For items, we have the following errors rates 1.62 and 5.62 for next event time and remaining time respectively. In the case of remaining time predictions, we observe improvement on error rates with the additional LSTM layer compared to iterative predictions proposed in [17]. We can see that the MAE for item object type is improved up to 10 times with our architecture.

Discussion. Based on the experimental results of Table 5, we get overall improvements for enriched event log in terms of accuracy for next activity prediction and MAE for time predictions. This can be explained by the addition of a numerical feature (number of items) to the input matrix, this feature can help extract the influence of the number of related items in the scheduling of activities and the time remaining until the end of the sequence. The exception for this is the

remaining time for the orders, where the enrichment of the log do not seem to improve the results. This may be explained by the fact that Orders is the global process for the considered log, thus already containing all the information about the related objects.

The model with an additional LSTM layer used for remaining time prediction has proven to be more efficient and accurate compared to the iterative model. Based on the experimental results, the proposed approach has enhanced the prediction performances on almost all the object types and event logs.

The proposed approach can be resumed in two steps. First, we tried feature enrichment using the number of related items which has proven experimentally to enhance the model's performances especially on the accuracy for next activity prediction. Second, we added a new LSTM layer dedicated to predict remaining sequence time instead of the classic iterative approach, this change has enhanced the prediction performances and reduced greatly the MAE on almost all object types and event logs both single and enriched. More experiments should be done to validate these preliminary results.

6 Conclusion

In this paper we proposed a first approach for predictive monitoring tailored to object centric logs. Specifically, we addressed the problem of next activity prediction, next event time and remaining time (until the end of process) prediction. A single neural network architecture based on LSTM has been experimentally evaluated for these tasks. We showed that taken into account related objects improve overall its performance. Compared with existing LSTM-based approaches for predictive monitoring, our architecture allows also to predict the remaining time until the end of the instance execution. The experimental results showed that the produced output for remaining time is better than the one computed by iteratively predicting next event time until the end of the process. In future work, we plan to propose new neural network architectures integrating more features of OCEL and compare them with the basic approach proposed in this paper. Our evaluation is based on the single OCEL log publicly available. We intend to extend this evaluation by searching more real-life logs from industrial partners or by building synthetic logs.

References

1. van der Aalst, W.: Object-Centric Process Mining: Dealing with Divergence and Convergence in Event Data, pp. 3–25 (09 2019). https://doi.org/10.1007/978-3-030-30446-1_1
2. van der Aalst, W.: Federated process mining: Exploiting event data across organizational boundaries. pp. 1–7 (09 2021). https://doi.org/10.1109/SMDS53860.2021.00011
3. van der Aalst, W., Adriansyah, A., Medeiros, A.K.A.d., Arcieri, F., Baier, T., Blickle, T., Bose, J.C., Brand, P.v.d., Brandtjen, R., Buijs, J., et al.: Process mining

manifesto. In: International conference on business process management. pp. 169–194. Springer (2011)
4. van der Aalst, W.M., Berti, A.: Discovering Object-centric Petri Nets. Fundamenta Informaticae **175**(1-4), 1 – 40 (2020)
5. Adams, J.N., Van Der Aalst, W.M.: Precision and fitness in object-centric process mining. In: 2021 3rd International Conference on Process Mining (ICPM). pp. 128–135 (2021). https://doi.org/10.1109/ICPM53251.2021.9576886
6. Bukhsh, Z.A., Saeed, A., Dijkman, R.M.: Processtransformer: Predictive business process monitoring with transformer network (2021). https://doi.org/10.48550/ARXIV.2104.00721, https://arxiv.org/abs/2104.00721
7. Camargo, M., Dumas, M., González-Rojas, O.: Learning Accurate LSTM Models of Business Processes, pp. 286–302 (07 2019). https://doi.org/10.1007/978-3-030-26619-6_19
8. Di Mauro, N., Appice, A., Basile, T.M.A.: Activity prediction of business process instances with inception cnn models. In: Alviano, M., Greco, G., Scarcello, F. (eds.) AI*IA 2019 – Advances in Artificial Intelligence. pp. 348–361. Springer International Publishing, Cham (2019)
9. Galanti, R., de Leoni, M., Navarin, N., Marazzi, A.: Object-centric process predictive analytics (2022). https://doi.org/10.48550/ARXIV.2203.02801, https://arxiv.org/abs/2203.02801
10. Ghahfarokhi, A.F., Park, G., Berti, A., van der Aalst, W.: Ocel standard (2020), http://ocel-standard.org/1.0/specification.pdf
11. Harl, M., Weinzierl, S., Stierle, M., Matzner, M.: Explainable predictive business process monitoring using gated graph neural networks. Journal of Decision Systems pp. 1–16 (06 2020). https://doi.org/10.1080/12460125.2020.1780780
12. Hinkka, M., Lehto, T., Heljanko, K.: Exploiting event log event attributes in rnn based prediction. In: Ceravolo, P., van Keulen, M., Gómez-López, M.T. (eds.) Data-Driven Process Discovery and Analysis. pp. 67–85. Springer International Publishing, Cham (2020)
13. Li, G., de Carvalho, R.M., van der Aalst, W.M.P.: Automatic discovery of object-centric behavioral constraint models. In: Abramowicz, W. (ed.) Business Information Systems. pp. 43–58. Springer International Publishing, Cham (2017)
14. Lin, L., Wen, L., Wang, J.: MM-Pred: A Deep Predictive Model for Multi-attribute Event Sequence, pp. 118–126 (05 2019). https://doi.org/10.1137/1.9781611975673.14
15. Pasquadibisceglie, V., Appice, A., Castellano, G., Malerba, D.: Using convolutional neural networks for predictive process analytics. In: 2019 International Conference on Process Mining (ICPM). pp. 129–136 (2019). https://doi.org/10.1109/ICPM.2019.00028
16. Philipp, P., Jacob, R., Robert, S., Beyerer, J.: Predictive analysis of business processes using neural networks with attention mechanism. In: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC). pp. 225–230 (2020). https://doi.org/10.1109/ICAIIC48513.2020.9065057
17. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with lstm neural networks. pp. 477–492 (05 2017). https://doi.org/10.1007/978-3-319-59536-8_30