Exploring Interpretable Predictive Models for Business Processes

Renuka Sindhgatta, Catarina Moreira, Chun Ouyang, and Alistair Barros
Queensland University of Technology
{renuka.sr, catarina.pintomoreira, c.ouyang, alistair.barros}@qut.edu.au

Abstract. There has been a growing interest in the literature on the application of deep learning models for predicting business process behaviour, such as the next event in a case, the time for completion of an event, and the remaining execution trace of a case. Although these models provide high levels of accuracy, their sophisticated internal representations provide little or no understanding about the reason for a particular prediction, resulting in them being used as black-boxes. Consequently, an interpretable model is necessary to enable transparency and empower users to evaluate when and how much they can rely on the models. This paper explores an interpretable and accurate attention-based Long Short Term Memory (LSTM) model for predicting business process behaviour. The interpretable model provides insights into the model inputs influencing a prediction, thus facilitating transparency. An experimental evaluation shows that the proposed model capable of supporting interpretability also provides accurate predictions when compared to existing LSTM models for predicting process behaviour. The evaluation further shows that attention mechanisms in LSTM provide a sound approach to generate meaningful interpretations across different tasks in predictive process analytics.

Keywords: interpretable models · attention-based neural networks · predictive process models

1 Introduction

Recurrent neural networks (RNNs) have been successful in tasks that require the processing of sequential data where, predicting the next output depends on an input sequence, such as language modelling, speech recognition or time series prediction [18]. RNNs and their variants such as Long short term memory networks (LSTM) have naturally found applicability in predictive process analytics by modelling the sequential business process execution data to predict the next event in a case, remaining execution time of a running case, and remaining sequence of events [1, 3, 7, 16, 15]. Accuracy has been a dominant criterion when choosing deep learning models (such as RNNs) for predictive process analytics. The use of these sophisticated models however, comes at the cost of the models being used as ‘black boxes’; i.e. they are unable to provide insights into why a
certain decision or prediction was drawn. Consequently, it is hard for users to understand the rationale of the black-box machinery when using predictions of the model for decision support. The opaqueness of these models often leads to lack of trust and impedes their adoption in sensitive domains such as insurance, healthcare, or law.

Recent literature has emphasised the need to understand and trust predictions from machine learning models coupled with a clear understanding of the behaviour of predictive models [13]. In parallel, there has been an increasing interest in the research community on interpretable or explainable machine learning [5]. Interpretability or explainability has not been addressed thus far in the context of predicting business process behavior when using deep learning techniques. We aim to address this gap in our work by building an attention-based LSTM model that provides insights into the important features influencing the prediction. Given an incomplete prefix representing a running case, the proposed model predicting the next activity can provide intuitive information on specific step or index in the sequence of events that influenced the prediction. For example, given a sequence of events, the model interpretation can reveal the influence of the last event or the first event on the prediction of next activity. In addition, it can reveal which attribute of an event (activity, role of a resource, or time) influenced the prediction. As a result, we present an approach to address the ‘black-box’ limitation using a two-level neural attention model that provides transparency of the relevant features influencing the prediction results while retaining the prediction accuracy compared to existing work on predicting business process behaviour using deep learning models. Hence, the paper makes the following contributions:

– proposes a model that facilitates interpretability in the context of predicting business process behaviour.
– presents experimental evaluation of the proposed approach and compares it to existing baselines thus addressing accuracy and interpretability.

The paper is organised as follows. A brief overview of previous studies using deep learning techniques to predict process behaviour is presented along with an introduction to the background topics such as interpretability and explainability (Section 2). The details of our approach to building interpretable machine learning models is presented in Section 3. The evaluation of the approach and the discussions on real-world event logs is presented in Section 4. Finally, we summarise the contributions of our work and outline future work (Section 5).

2 Related Work and Background

2.1 Related Work

The use of deep learning techniques in the form of RNNs and its variants for predicting process behaviour has been addressed in the literature. Layers of stacked LSTM have been used to predict next activity of an ongoing case and the time
of a case until completion (or remaining time) [15, 16]. The input to the model is a sequence of prior events. Each event is encoded to a feature representing the activity of the event as a one-hot vector, and additional dimensions representing the timestamp of the event. The architecture comprises a number of shared LSTM layers, and independent LSTM layers. The number of shared and independent layers are configured to achieve the best performance in terms of the accuracy of predicting the next activity and mean absolute error in predicting the time of the next event. The remaining sequence of events (suffix) of a running case is generated by iteratively predicting the next event until the end event is predicted.

Evermann et al. [3] proposed a deep learning model with two layers of stacked LSTM to predict the next event of an ongoing case. The categorical attributes of the event such as activity and resource information are transformed to n-dimensional vector or embedding. The effect of the dimensionality of embedding and the length of the input sequence on accuracy of the predictions are evaluated. This study acknowledges the limitation of interpreting the knowledge encoded by the model. The study further presents two mechanisms of interpreting the results of the model: i) using hallucinations to predict the suffix, and ii) using t-SNE plots to visualise the embedding matrix and the hidden states. The interpretations provide some information about the layers and cells activated for an input sequence but are limited in their ability to explain a prediction.

Lin et al. [7] propose an encoder, modulator and decoder model to predict the next event and generate the suffix of a case. The encoder comprises of an LSTM network for each event attribute. The modulator combines the hidden representations of the LSTM-based encoder to infer the weight vector. The weighted sum of the encoded vectors is used as input to a decoder layer comprising of a two layer LSTM network to predict the next activity. While the modulator computes a weight vector that could represent importance of different event attributes and provide insights into which attribute is important for the prediction, the focus of the study has been on the accuracy of the proposed architecture.

Recent work by Camargo et al. [1] presents an approach to train accurate LSTM-based models to predict process behaviour. The approach consists of a pre-processing phase that extracts n-grams or fixed length sequences of events, an LSTM training phase, and a post-processing phase to select the predicted next event. The majority of state-of-the-art research focus uniquely on the accuracy of the models. The focus of this work is to infuse some form of interpretability directly into an LSTM-based predictive model predicting business process behaviour without impacting accuracy.

### 2.2 Interpretable Models for Predicting Process Behaviour

Existing state-of-the-art techniques for constructing predictive models for business processes usually adopt a “black-box” approach. Black-box models commonly refer to the use of machine learning models where the internal mechanisms of the models are either unknown or known but not understandable to users. The challenge is to endow complex models with capabilities to explain
the underlying predictive mechanisms in a way that helps users understand and scrutinize the the behaviour. To achieve interpretability, the body of literature is divided into two major paradigms of approaches towards interpretability of black-boxes: (1) interpretable models; and (2) post-hoc models.

- **Interpretable models** are interpretable by design, promoting a more transparent, white-box approach for prediction. An interpretable model is capable of explaining a decision it takes or explaining how it works [5]. Examples include decision trees, Bayesian networks, linear regression. These systems enable an understanding of how features correlate with each other and how they contribute to the predictions.

- **Post-hoc models**, which are model-agnostic, aim to provide local explanations for a specific decision and make it reproducible on demand (instead of explaining the whole model’s behaviour). The two most representative post-hoc models in the literature are LIME [12] and SHAP [8], and they are based on two completely different mechanisms. While LIME makes use of feature perturbations to build a linear surrogate model out of it (such as a decision-tree), SHAP is based on game theory, where predictions are explained by assuming that each feature value of the instance is a “player” in a game where the prediction is the payout. SHAP makes use of Shapley values, which is a method to fairly distribute the “payout” among different features [9].

When it comes to extracting interpretations out of deep learning models such as RNNs and LSTM, concerns have been raised on post-hoc explanation models as these explanation methods can be an inaccurate representation of the original model [13]. Although several approaches exist in the literature to probe and interpret deep neural networks [2, 4, 6, 10], when it comes to interpretable models within predictive process analytics, the literature proposals are limited and their efficiency has not been thoroughly analysed. Recent research has acknowledged the need for extracting interpretations by illustrating the potential of explainable models for a manufacturing business process [11].

### 2.3 Attention-Based Models for Interpretations

Attention-based models can be seen as neural network architectures that enable the learning of sequential input-output relations when the input and output sequences have different lengths. This type of structure is of particular interest to predictive process behavior, because traces have different lengths. While existing predictive process analytics methods in the literature compress the sequential input into a fixed sized vector in order to feed it to a neural network architecture, attention-based models can use variable-length inputs without the constraint of fixed-sized vector by using a variable-length memory. In general, the attention mechanism allows a predictive model to focus on (or attend to) specific elements in a given input sequence for a prediction task.

Attention-based models also provide an underlying mechanism for interpretability [14]. Since these models require the computation of a distribution
of weights over inputs, the attention weights can provide some insights to a
decision-maker of why a certain prediction was computed by means of not ma-
nipulating directly the input features (as it happens in LIME), but rather by
manipulating the distribution of weights, which are associated to the input [14].

Current studies in the literature have used attention-based models mainly
in Natural Language Processing tasks [17, 6, 4]. For a model to be interpretable,
it must not only suggest explanations that make sense to the user, but also
ensure that those explanations accurately represent the true reasons for the
model’s decision [14]. This is, however, nearly impossible to verify in traditional
post-hoc models, such as LIME and SHAP, since these algorithms build models
around local interpretations, providing approximations to the predictive black
box, instead of reflecting the true underlying mechanisms of the black box (as
pointed out by [13]). With attention-based models, since the interpretability is
extracted directly from the weights that are used to train the black box, one can
get interpretations that better reflect the underlying mechanisms of the black
box. In the next section, we present how attention models can be used as an
interpretable mechanism in the context of predictive process analytics.

3 Approach

This section details our approach to predict process behaviour. We first describe
the input features used for our model. Next, we describe the neural network
architectures, which extend from models based on reverse and dual attention
mechanisms [2, 10].

**Input Features:** The objective of the model is to predict the next activity
given the trace of an incomplete case. A trace is a sequence of events of a case.
The input to the model are multiple prefix traces of different lengths. A prefix
trace of length \( k \) contains the first \( k \) events of a trace. For simplicity, we describe
the model input for a single prefix trace.

An event \( e_1 \) in a prefix trace is represented as \((a_i, r_l, t_i)\), where \( a_i \in A \) is
the activity of the event, \( r_l \in RL \) is the role of the resource performing
the activity, and \( t_i \in \mathbb{R} \) is the timestamp associated with the event. We assume that
each event has a timestamp associated with the completion of the activity. For
each event, we represent the activity vector as a binary vector \( v_{a_i} \in \{0, 1\}^{|A|} \)
where \( v_{a_i,p} \) is set to 1, if \( p \) is the activity of the event and the rest are set
to 0. Similarly the role vector is a binary vector \( v_{rl_i} \in \{0, 1\}^{|RL|} \) with the
role of resource associated to the event set to 1 and 0 otherwise. The time
feature is computed as the time elapsed between the completion time of the
event and that of its previous event. The time feature for event \( e_i \) is computed
as \( \Delta t_i = t_i - t_{i-1} \). Fixed length sequences are extracted from each trace as
detailed in the Section 4, in line with the previous work [1]. Each fixed length
sequence of events \((e_1, e_2, \ldots, e_i)\) has an event represented by an activity vector,
a role vector, and a continuous time interval value \((v_{a_i}, v_{rl_i}, \Delta t_i)\).
Attention-based LSTM models: The section presents an overview of the three models used in this work: i) prefix-index attention model that provides insights into the events influencing the prediction, ii) prefix-index and event attribute attention model that enables interpreting the events as well as the input features (representing event attributes) influencing the prediction, and iii) an attention-based interpretable model to predict next activity and its completion time. The model architectures are based on an interpretable model used in the health-care domain to predict patients at risk [2].

Prefix-index attention model: Figure 1(a) presents the model that predicts the next activity of a running case, given an input prefix of events. The model takes as input, a fixed length sequence of activity vectors, role vectors along with sequences of time intervals. An embedding layer is used to convert the categorical values of the activity and the role represented as binary vectors to continuous vectors. The activity embedding, role embedding, and the time interval value are concatenated as $v_{emb}$. The concatenated input representing the sequence of events $(v_{emb1}, v_{emb2}, \ldots, v_{embi})$ is passed to a bidirectional LSTM. Compared to a traditional LSTM which processes sequence information in one direction or from the first input in the sequence to the last input, a bidirectional LSTM computes additional set of hidden state vectors by considering reverse order of the input. The reverse order is useful to consider in our scenario, as in a case execution, the next activity relies on the information about what has happened previously (reverse order of activities). The output of the BiLSTM is a
hidden state vector \((h_1, h_2, \ldots, h_i)\) for each index in the sequence. The attention layer takes as input the hidden state vectors and generates the index attention \((\alpha_1, \alpha_2, \ldots, \alpha_i)\), a distribution of values that sum to 1. The attention values are element-wise multiplied with the concatenated input vector and added to obtain a context vector \(c_i = \sum_{j=1}^{i} \alpha_j \odot v_{emb_j}\), where \(\odot\) indicates element-wise multiplication. The context vector is used as the input to a neural network dense layer to predict the next activity. The attention mechanism allows the model to focus on specific indices in the prefix when predicting the next activity. Finding the indices in a prefix that contributes to a prediction can be derived based on \(\alpha\) values - the higher the value, the more influential the event at that index was in predicting the next activity. Hence, the model architecture enables interpreting the indices and thus the events influencing a prediction.

**Prefix-index and event attribute attention model:** In addition to identifying the events in a prefix influencing the prediction, it would also be useful to reason which of the event attributes influenced the prediction (activities, resource roles, or the previous execution time). To support event attribute-level attention, another BiLSTM layer is used and hidden vectors are computed (Figure 1(b)). The variable-level attention weights \(\beta\) are derived from the hidden vectors of the BiLSTM layer. The details on how to achieve variable-level attention can be found in [2].

In brief, the context vector \(c_i\) is computed using the \(\alpha\) weights, the \(\beta\) weights, and the input vector embeddings. The context vector, \(c_i = \sum_{j=1}^{i} \alpha_j \beta_j \odot v_{emb_j}\). If there are \(m\) activities and \(r\) roles, the input to the BiLSTM layer representing an event is an \((m + r + 1)\) dimension vector (including the time interval dimension). The \(\beta\)-weight for each event is an \((m + r + 1)\) vector. As the input features are binary vectors, the context vector indicates the influence of each input dimension on the prediction. For activities and roles, the \(\alpha \beta \odot v_{emb}\) weight itself represents its influence on the prediction. For non-binary time interval input, the weight \(\alpha_j \beta_j \times \Delta t_j\) is able to provide insight of the contribution of the time interval for predicting the next activity.

**Prefix-index and event attribute attention model for remaining time prediction:** The prefix-index and event attribute attention-based model is used to predict next activity, role and remaining time (Figure 2). The model is used to compare the accuracy of attention-based model with existing approaches. In this model, the input features remain the same. The output context vector is used as input to three dense layers. The model shares the context vector. The next activity, the resource role and the time interval are predicted by the three dense layers. This model is further used to generate the complete trace. This is achieved by injecting continuous feedback to the model for each new predicted event, until the end of the trace (stop event) is reached. The time interval for each event is predicted and the remaining time for case execution can be computed as a sum of the time intervals.
4 Evaluation and Results

This section describes the datasets used, the pre-processing of the input event attributes, and three experimental evaluations. The first experiment evaluates the prefix-index attention model for accuracy and uses the $\alpha$ attention weights to interpret the influential events. The second experiment evaluates prefix-index and event attribute attention model to interpret the influence of the event attributes on the prediction. We finally compare the performance of the attention-based model with existing baseline approaches.

4.1 Datasets and Processing

The experiment uses five real-world business process event logs available at the 4TU center for research data\(^1\). The distinct characteristics of the event logs is presented in Table 1. The logs represent traces of short lengths (Helpdesk and BPIC 2013) with an average of 5 events per trace, to long running cases with with 20-30 events per trace (BPIC 2012 and BPIC 2015-5).

**Input sequence extraction:** The activity, resource, and completion time of the event are used as input features. Resources are grouped into roles based on the work by Camargo et al. [1] and the role of the resource associated to the event is used as the input feature. The categorical attributes of the event, i.e. activity and role are one-hot encoded (i.e. represented as a binary vector). The use of one-hot encoding improves interpretability as described in Section 3. Fixed length sequences are generated for each trace by extracting n-grams. An example of n-grams of length 5 extracted for a trace containing the following sequence of activities [A,B,C,F,K,R] is shown in Table 2. For each trace,

\(^1\) https://data.4tu.nl/repository/collection:event_logs_real
a start and end activity is added before the extraction of n-grams. A similar approach is used to generate the n-grams of roles associated to the events in the trace and the time intervals of the events. An n-gram of size $k$ effectively constitutes $k$ prior events of the case used to predict the next activity.

The **time interval** input is a continuous value. We scale continuous values to its z-score (i.e. zero mean and unit variance) resulting in values in the $[-1,1]$ range. This is a standard normalisation approach used by the deep learning implementations.

**Training details:** For each event log, we temporally split into train set and test dataset in a 0.7 : 0.3 ratio. The training set consists of the first 70% of the cases. The n-grams are extracted for the cases in the train set. The train set is further split into training and validation sets in a 0.85 : 0.15 ratio. The validation set is used to determine the values of the hyper-parameters including the number of LSTM cells, and the regularisation parameters.

### 4.2 Prefix-index attention model

Table 3 summarises the accuracy of the prefix-index attention model for different logs and different n-gram lengths. The length of the n-gram is chosen based on the average and median number of activities per trace. For the Helpdesk, BPIC 2013 log, n-gram length is chosen to be 5 as the average number of events per trace is low ($<5$). For longer traces, the n-gram with different lengths are evaluated. Table 3 indicates that the accuracy of the models does not significantly change with the increase in length of the n-gram. We use the attention weights to gain insights into the prediction accuracy of the models. Figure 3 plots the $\alpha$-attention weights for the 8 models presented in Table 3. The x-axis is the index of the n-gram or the input sequence of events (ranging from 0 to n-gram length). The $\alpha$-weight is computed for each prediction in the test dataset (30% of unseen data). The y-axis is the average $\alpha$-weight at each index. Figure 3 shows that the

| Event Log   | # traces | # activities | mean events/trace | median events/trace | mean duration | median duration |
|-------------|----------|--------------|-------------------|---------------------|---------------|-----------------|
| Helpdesk    | 4580     | 14           | 4.6               | 4                   | 40.9 days     | 39.9 days       |
| BPIC 2013   | 13087    | 7            | 4.4               | 3                   | 179.2 days    | 82 days         |
| BPIC 2012-W | 9685     | 6            | 7.5               | 6                   | 11.4 days     | 8.5 days        |
| BPIC 2012   | 13087    | 36           | 20                | 11                  | 8.6 days      | 19.5 hours      |
| BPIC 2015-5 | 1156     | 41           | 31.3              | 31                  | 98.3 days     | 77.1 days       |

Table 1. Event logs and the summary statistics
| No. | Event Log    | n-gram length | Accuracy |
|-----|--------------|---------------|----------|
| 1   | Helpdesk     | 5             | 0.7922   |
| 2   | BPIC 2013    | 5             | 0.4609   |
| 3   | BPIC 2012-W  | 5             | 0.7682   |
| 4   | BPIC 2012-W  | 10            | 0.7789   |
| 5   | BPIC 2012    | 5             | 0.7900   |
| 6   | BPIC 2012    | 10            | 0.7900   |
| 7   | BPIC 2012    | 15            | 0.7904   |
| 8   | BPIC 2015-5  | 15            | 0.3682   |

Table 3. Accuracy predicting next event with different n-gram size

α-weight for the last event is 0.80 for the models trained using the Helpdesk, and BPIC 2013 event logs indicating that these models predominantly use the last event to predict the next event. The influence of the last event is lower in other models as indicated by α-weights of the last index (\(<= 0.5\) for BPIC 2012, BPIC 2012-W, and BPIC 2015-5). We observe that with BPIC 2015-5, the average α-weights of the last state is low with all prior indices having similar weights. The model accuracy is also low (0.35). Here, we hypothesise that with less training data (550 traces), the model is unable to learn and the attention layer has not yet converged to a stable set of weights.

To validate the influence of α-weights, we train models on all event logs by considering only the last two events; i.e. an input n-gram length of 2 only. For models that indicated high α-weights for the last two events (\(> 0.8\)), we expect that the accuracy of predictions should continue to remain high if we reduce the n-gram size to 2. Table 4 shows the accuracy of the models trained using lower input sequence lengths. Models trained on Helpdesk, BPIC 2013 have the same accuracy as shown in Table 3. The model trained on BPIC 2012-W reduces moderately. The accuracy lowers considerably for models trained on BPIC 2012 and BPIC 2015-5, as considering only last two events results in loss of input information; i.e. the sum of α-weights of the last two events is lower than 0.7 with prior events influencing the prediction of next event. The results validate the interpretation provided by α-weights thus providing information about the events influencing the prediction.

| No. | Event Log    | n-gram length | Accuracy |
|-----|--------------|---------------|----------|
| 1   | Helpdesk     | 2             | 0.7930   |
| 2   | BPIC 2013    | 2             | 0.4633   |
| 3   | BPIC 2012-W  | 2             | 0.7604   |
| 4   | BPIC 2012    | 2             | 0.7261   |
| 5   | BPIC 2015-5  | 2             | 0.3295   |

Table 4. Accuracy predicting next event
Fig. 3. The mean distribution of $\alpha$-weights for the index values of the test dataset predictions with n-gram index (x-axis) in the range of [5,15] when predicting next activity.

4.3 Prefix-index and event attribute attention model

The prefix-index and event attribute attention model is trained to support interpretations on the most influential events ($\alpha$-weights) in a running case and
the event attributes attention-weights \((\alpha \beta \odot \nu_{emb})\) that contribute to the prediction. The models are trained on all five event logs. For each prediction in the test dataset, the most influential index in the sequence of inputs is identified, i.e. the input index with the highest \(\alpha\)-weights. The \(\beta\)-weights of the input attributes are also computed. The mean attention weights are presented for all five event logs in Figure 4. The attention weights are in the range \([-1.0, 1.0]\). The absolute value of the weight is important as both positive and negative values indicate the influence of an attribute on the prediction. To understand the attributes influencing the prediction using attention weights, we further perform an ablation study of the time and resource features. The basic idea of an ablation is to remove features systematically and investigate how a feature impacts the prediction task. The attention weights in Figure 4 indicate that time interval feature does not influence the prediction of next activity for models trained on the event logs BPIC 2012, BPIC 2015-5, Helpdesk. Hence, in these cases, not including time as an attribute would not reduce the accuracy. As expected, we observe minimal impact on accuracy when the time feature is excluded (AR (-T)) on

Fig. 4. The mean distribution of attention-weights for the attribute values for prefix-index and event attribute attention model when predicting next activity.
models trained using these logs in Table 5. The resource role feature influences the prediction as indicated by attention weights. Removal of resource role as a feature (A (-R -T)) reduces accuracy of all models. The influence is higher on models where there are more number of roles with higher attention weights (e.g. BPIC 2012) as compared to models with lower number of role values (e.g. Helpdesk).

| No. | Event Log   | ART   | AR (-T)  | A (-R -T)  |
|-----|-------------|-------|----------|------------|
| 1   | Helpdesk    | 0.7961| 0.7855   | 0.7829     |
| 2   | BPIC 2013   | 0.4648| 0.4638   | 0.4598     |
| 3   | BPIC 2012-W | 0.7741| 0.7697   | 0.7684     |
| 4   | BPIC 2012   | 0.7874| 0.7841   | 0.7625     |
| 5   | BPIC 2015-5 | 0.3624| 0.3581   | 0.3551     |

Table 5. Activity (A), Role of the Resource(R), Time(T) feature ablation on the prefix-index and event attribute attention model

Generally, the prefix-index and event attribute attention model provides richer insights for the same n-gram length vis-a-vis the prefix-index attention model. The attention weights offer information about the influential events and the event attributes while maintaining the same or better accuracy when compared to the prefix-index attention model.

![Attention weights](attachment:image.png)

**Fig. 5.** The attention weights: prefix index attention-weight, and event attribute attention-weights for a single prediction.

**Local Interpretations:** The attention weights presented so far are global interpretations or interpretations that have been aggregated (we use arithmetic mean for aggregation). However, the model provides local interpretations or is able to provide the reasoning for a particular prediction. Figure 5 presents the attention weight for a single input prefix for model trained using Helpdesk event log. For this specific instance of an input, the model considers the last three events
and further uses time interval as an important feature when predicting the next event.

### 4.4 Accuracy and Interpretability of models

The aim of training the attention-based activity and remaining time prediction model is to assess the performance of the model when predicting the next event, the remaining sequence of events (i.e. suffixes), and the remaining time given the input prefixes of varying lengths. For each prefix, the next activity is predicted and the accuracy is measured. For suffix and remaining time prediction, the *hallucination* approach outlined by earlier studies [1, 3] is used till the end of the case is reached. The model predicts the remaining time $\Delta t_{i+1}$, which is the difference between the timestamp of the last event in the prefix from the timestamp of the last hallucinated event. The Damerau–Levenshtein (DL) measure (which applies to pairs of strings), is used to measure the similarity for suffix prediction [1]. The Mean Absolute Error (MAE) metric is used to measure the error in predicting remaining time. MAE is computed by taking the absolute value of the difference between the true value of remaining time and the predicted value, and then calculating the average value of these magnitudes.

| Implementation      | Next event | Suffix prediction distance | Remaining cycle time MAE (days) |
|---------------------|------------|----------------------------|--------------------------------|
|                     | Helpdesk   | BPIC 2012                  | BPIC 2012-W                    | Helpdesk | BPIC 2012 | BPIC 2012-W | Helpdesk | BPIC 2012 |
| Our approach        | 0.796      | 0.787                      | 0.774                          | 0.910     | 0.319     | 0.467       | 8.12     | 9.3        |
| Camargo et al.      | 0.789      | 0.786                      | 0.778                          | 0.917     | 0.632     | 0.523       | 6        | 9.1        |
| Evermann et al.     | 0.798      | 0.780                      | 0.623                          | 0.742     | 0.110     | 0.297       | -        | -          |
| Tax et al.          | 0.712      | 0.760                      | 0.767                          | 0.353     | 6         | 9.1         | -        | -          |
| Lin et al.          | 0.912      | 0.974                      | 0.874                          | 0.281     | -         | -           | -        | -          |

**Table 6.** Next event, suffix, and remaining time prediction performance

We use the Helpdesk, BPIC2012W and BPIC2012 event logs evaluated in previous studies. Table 6 summarizes the average accuracy for the next-event prediction task, the average similarity between the predicted suffixes and the actual suffixes, and the MAE of the remaining time. For the task of next-event prediction, our approach performs similar to all existing state-of-the-art approaches [1, 3, 15] except for the baseline of Lin et al. [7]. For the task of suffix prediction, our approach has the performance similar to Camargo et al [1] for suffix prediction on Helpdesk log, but underperforms with the other logs. Our approach has higher MAE for the remaining time prediction as compared to Camargo et al. While the focus of our work has been to improve interpretability of the models, these results suggest that the model is able to achieve good performance when predicting categorical variables. The model requires further optimisation.
and tuning with respect to predicting the continuous value of remaining time. However, we observe that the approach presented in this work leads to deriving interpretations from the models without a significant trade-off on accuracy when predicting process behaviour.

5 Conclusion and Future Work

This paper presents an attention-based LSTM model that provides interpretations to predict the next event and its attributes: event activity, the role of the resource, and the timestamp associated with the event. The attention mechanisms provide insights into the events in a case and the event attributes that influence prediction of the next event. We use the model to generate the suffix and estimate remaining time by iteratively predicting the next event until the end of the case is reached. To process the input, we continue to use the one-hot encoding for categorical data but convert the binary vectors to continuous vectors. The paper presents three network architectures. Firstly, the prefix-index attention model supports interpretations regarding which events influenced a prediction. Secondly, prefix-index and event attribute attention model identifies which events and event attributes influenced the prediction. Finally, we extend the model to support the prediction of three event attributes: the activity, role of the resource and its timestamp.

In this work, we have limited the input features to three event attributes by considering the activity, resource and time perspective. The data perspective and inclusion of other generic features has not been considered. Additional process attributes can have a significant impact when predicting the next event for certain business processes. We would extend the current model to include the data perspective as a part of the future work. Our work provides a starting point for building and evaluating models that explain their predictions. Providing insights to business users on the predictions can be a very challenging task and remains as an open research question in the scientific community [13].

Acknowledgement: We particularly thank Manuel Camargo, Marlon Dumas, and Oscar González Rojas for the high quality code they released which allowed fast reproduction of the experimental setting and the processing of event logs.

Reproducibility: The source code and the event logs can be downloaded from https://git.io/JvSWI

References

1. Camargo, M., Dumas, M., Rojas, O.G.: Learning accurate LSTM models of business processes. In: 17th International Conference, BPM. pp. 286–302 (2019)
2. Choi, E., Bahadori, M.T., Sun, J., Kulas, J., Schuetz, A., Stewart, W.F.: RETAIN: an interpretable predictive model for healthcare using reverse time attention mechanism. In: Annual Conference on NeurIPS. pp. 3504–3512 (2016)
3. Evermann, J., Rehse, J., Fettke, P.: Predicting process behaviour using deep learning. Decis. Support Syst. 100, 129–140 (2017)
4. Ghaeini, R., Fern, X.Z., Tadepalli, P.: Interpreting recurrent and attention-based neural models: a case study on natural language inference. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (2018)
5. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comput. Surv. 51(5), 93:1–93:42 (Aug 2018)
6. Lee, J., Shin, J.H., Kim, J.S.: Interactive visualization and manipulation of attention-based neural machine translation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing: System Demonstrations (2017)
7. Lin, L., Wen, L., Wang, J.: Mm-pred: A deep predictive model for multi-attribute event sequence. In: Berger-Wolf, T.Y., Chawla, N.V. (eds.) Proceedings of the 2019 SIAM International Conference on Data Mining, SDM. pp. 118–126. SIAM (2019)
8. Lundberg, S.M., Lee, S.I.: A unified approach to interpreting model predictions. In: Proceedings of the 31st Conference on Advances in Neural Information Processing Systems (NIPS) (2017)
9. Molnar, C.: Interpretable Machine Learning: A Guide for Making Black Box Models Explainable. Leanpub (2018)
10. Qin, Y., Song, D., Chen, H., Cheng, W., Jiang, G., Cottrell, G.W.: A dual-stage attention-based recurrent neural network for time series prediction. In: IJCAI. pp. 2627–2633 (2017)
11. Rehse, J., Mehdiyev, N., Fettke, P.: Towards explainable process predictions for industry 4.0 in the diki-smart-lego-factory. KI 33(2), 181–187 (2019)
12. Ribeiro, M.T., Singh, S., Guestrin, C.: “Why should I trust you?”: Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD. pp. 1135–1144 (2016)
13. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nature Machine Intelligence 1(5), 206–215 (2019)
14. Serrano, S., Smith, N.A.: Is attention interpretable? In: Proc. of the 57th Conference of the Association for Computational Linguistics, ACL. pp. 2931–2951. Association for Computational Linguistics (2019)
15. Tax, N., Verenich, I., Rosa, M.L., Dumas, M.: Predictive business process monitoring with LSTM neural networks. In: Advanced Information Systems Engineering - 29th International Conference, CAiSE 2017, Proceedings. Lecture Notes in Computer Science, vol. 10253, pp. 477–492 (2017)
16. Verenich, I., Dumas, M., Rosa, M.L., Maggi, F.M., Teinemaa, I.: Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring. ACM TIST 10(4), 34:1–34:34 (2019)
17. Wang, Y., Huang, M., Zhu, X., Zhao, L.: Attention-based lstm for aspect-level sentiment classification. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (2016)
18. Williams, R., Zipser, D.: A learning algorithm for continually running fully recurrent neural networks. Neural Computation 1, 270–280