Towards Accurate Predictions in ITSM: A Study on Transformer-Based Predictive Process Monitoring

Marc C. Hennig[1][0009-0005-6185-2623]

[1] University of Applied Sciences Munich, Lothstraße 64, 80335 Munich, Germany
mhennig@hm.edu

Abstract. The accurate prediction of service process performance, particularly in IT service management (ITSM), is critical for adhering to service-level agreements and avoiding associated penalties. However, existing predictive process monitoring solutions, predominantly based on recurrent neural networks, have been found to be inadequate in handling ITSM processes. Notably, the heterogeneity in process artifacts and environments impairs process predictions. This research proposes a novel transformer-based architecture to effectively handle IT service process event logs. By integrating advanced positional encoding techniques and distinguishing between static and dynamic attributes, a novel transformer architecture is evaluated using multiple publicly available ITSM event logs. This architecture demonstrates its potential to deliver more accurate predictions than LSTM models in terms of remaining time predictions. This work provides experimental results into the application of transformer architectures for predictive process monitoring, paving the way for enhanced efficiency in ITSM.

Keywords: predictive process monitoring, transformer, GLU, ITSM.

1 Introduction

In today's economy, the success of a service provider is increasingly tied to an organization's ability to provide its processes in time [1]. Ensuring an efficient execution of the underlying service processes is particularly relevant in this regard, as providers are bound by contractual service-level agreements (SLA) [1]. SLAs usually involve severe penalties in case of time or quality-related deviations, and adherence to them is an important performance indicator.

In this setting, service process instances benefit greatly from predictive insights of the expected remaining time to facilitate decision-making and avoid time-related SLA violations. Predictive process monitoring (PPM) has emerged as a tool for forecasting the future states of ongoing process instances [2]. However, IT service management (ITSM) processes, in particular, exhibit an inherent complexity due to heterogeneity in artifacts [3] and process environments [4]. Additionally, their ad-hoc nature and reliance on individual expertise complicate the accurate forecast of process instance runtimes and outcomes, often leading to results that are too inaccurate for practical use [4]. This underscores the need for innovative solutions in this area.

These challenges in ITSM processes have hindered the widespread adoption of PPM. Although many predictive process monitoring architectures have been created [2], only a few have been successfully applied to ITSM event logs. Given this limited application of predictive process monitoring in ITSM, there is a clear need for innovative approaches. This research addresses this need by designing a transformer architecture [5] that can effectively handle complex IT service process event logs. The transformer architecture [5] has replaced commonly used recurrent neural networks (RNN) and their derivatives for many sequence-related tasks [6]. Still, it has yet to gain significant traction in PPM compared to other domains [2]. It is therefore assessed in this work, attempting to answer the following research question: *How can a predictive process monitoring transformer architecture be designed to accommodate the intricacies of IT service processes?*

As part of a research project investigating how PPM can transform ITSM operations, this work's approach consists of three main parts. First, the literature regarding transformers in PPM in terms of their architecture and the different approaches taken is analyzed. Second, a transformer architecture is designed and implemented based on these insights. Finally, the model is applied to event logs from ITSM's incident management, and the results are quantitatively evaluated to assess how techniques new to PPM can be used therein, potentially outperforming other solutions. Due to its relevance for highly operational incident management [4], the focus lies on the remaining time prediction, contributing to a better understanding of transformers in PPM.

2 Research Method

In this work, design science research is applied, as described by [7], as the central framework with the steps outlined in Fig. 1. The research problem is analyzed by reviewing the ongoing research on transformers in PPM and innovations from other domains using transformers that might be transferable to PPM. A transformer architecture is then designed and developed based on the insights from the literature research, the special requirements, and the unique properties of ITSM processes outlined in the previous section. During the architecture development, the focus is explicitly laid on the input preparation and positional encoding in addition to providing a ground-up explanation of the different model components.

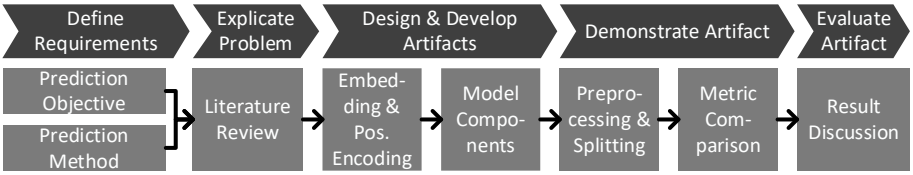

Fig. 1. Design science research framework following [7]

The results are subsequently demonstrated and experimentally evaluated using multiple publicly available event logs from ITSM, focusing specifically on applying a reproducible preprocessing setup using strict temporal splitting [8] to ensure comparable

results. The models are trained on the prepared event logs, and their performance is measured using regression metrics appropriate for the remaining time prediction. Finally, the results are compared with baseline solutions to allow an interpretation of the findings and the developed transformer architecture.

3 Research Background

PPM has traditionally depended extensively on RNNs, particularly Long-Short-Term Memory (LSTM) networks, which are commonly used for their capacity to handle sequential data effectively [2, 9]. However, other architectures and use cases have continuously evolved. This study focuses on the usage of transformers in PPM, which is motivated by both theoretical and practical benefits [5]. To achieve this, the existing literature on the application of transformers in PPM is first collected and analyzed. Transferable architectural features from other domains are then identified and contextualized within the findings of prior studies to identify possible research gaps.

3.1 Transformer in Predictive Process Monitoring

Most earlier works in PPM use simpler transformer architectures consisting primarily of single transformer encoders [10, 11] close to the original implementation [5]. The *ProcessTransformer* [10] is particularly notable as it is an often-used baseline model consisting of a transformer encoder and minimal preprocessing. Transformers have diversified in newer research, resorting to more specialized solutions. However, transformer encoders are still popular, while decoders remain a niche choice. Despite their popularity in natural language processing, models like BERT [12] and GPT-2 [13] are not widely used, possibly due to their low performance compared to later solutions [10].

Table 1. Overview of the work on PPM and transformers oriented on [2]

Ref.	Network Type	Seq. Encoding	Attr. Handling	Pos. Encoding	Target
[14]	Hybrid	BiLSTM + Attention	Dynamic	–	OUT
[11]	Transformer	Decoder	–	Fixed	NA
[15]	Hybrid	LSTM + Attention	–	–	NA, SEQ
[10]	Transformer	Encoder	–	Learned	NA, NT, RT
[13]	Transformer	Decoder (GPT2)	Dynamic	Learned	NA
[12]	Transformer	Encoder (BERT)	–	Learned	NA, OUT
[16]	Hybrid	BiLSTM + Attention	Dynamic	–	NA
[17]	Transformer	Encoder	Dynamic	Fixed	NA, ATTR
[18]	Hybrid	LSTM + Attention	Dynamic	–	NA
[19]	Transformer	Encoder	–	Custom	NA
[6]	Transformer	Encoder	Dynamic	Fixed, Learned	NA, RT
[20]	Transformer	Encoder	Dynamic	Learned	NA, NT, RT

The strategies for attaining improved prediction performances vary widely. Hierarchical architectures, either as a hybrid of RNNs with the transformer's central attention

mechanism [15, 16] or stacked attention over different granularities [6], outperformed several earlier solutions. This might be due to their improved ability to capture local process structures and long-range dependencies, which is hypothesized to improve model performance [21]. Additionally, the integration of further attributes might contribute to the performance [22]. While not hierarchical, other approaches have adopted multiple encoders [17], mainly to capture event-specific, dynamic attributes. Thus, these attributes are integrated as concurrent sequences in addition to the activity. Interestingly, while dynamic attributes are often included, case-specific, static attributes and their separate handling remain underexplored. They are often sequentialized and treated similarly to the dynamic attributes, despite positive indications for their separate handling from other domains [23].

The next activity (*NA*) classification is the most popular choice regarding the prediction targets with transformers, as seen in Table 1. In contrast, regression problems like the remaining time (*RT*) and next timestamp (*NT*) prediction are far less commonly approached. Predicting specific process instance outcomes (*OUT*) or other attributes (*ATTR*) is also less often performed, similar to sequence-to-sequence (*SEQ*) predictions [2]. Various transformer architectures have been developed, with a trend toward multiple attentions for dynamic attributes. This suggests a gap in the current research on investigating the benefits of distinctly integrating static attributes to enhance the prediction performance of PPM models.

3.2 Architectural Features in Transformer Models

Transformers are widely used in other domains, leading to several improvements that could benefit PPM. One key innovation is in positional encodings, where supplementary information beyond the absolute position of an element in a sequence enriches the data. While encoding static context information through additional domain knowledge has been described [19] in PPM, most works use fixed or learned positional encodings, as shown in Table 1. From outside PPM, timestamp encoding seems especially relevant and has succeeded in transformer-based time series predictions [24]. Also, it is currently suspected that relative encodings might outperform commonly used absolute encodings, leading to the development of combinations [25] to mitigate disadvantages.

Another essential aspect is converting transformer layer outputs from two-dimensional matrices into single vectors suitable for downstream prediction tasks. Standard techniques in PPM include average pooling layers [10, 20], weighted summation of outputs [16], and flattening [6]. In natural language processing, hidden states are often extracted based on a special token or the last position [12], but techniques vary widely.

Additionally, the enhanced performance of transformers using gated linear units (GLU) and gated residual networks (GRN) is notable. The *Temporal Fusion Transformer* [23], which employs these techniques to incorporate static attributes and enhance self-regularization, is a well-known example. In other studies, GLUs have also been shown to benefit transformers [26], suggesting their utility for PPM. None of these approaches have been applied to PPM, indicating an actionable research gap.

4 Model Development

Based on the analysis of the related research, it was identified that integration methods for static attributes are missing, especially, and that the use of GLUs and alternative positional embedding methods is under-researched. A transformer-based architecture is developed using these insights, focusing on incorporating static attributes separately from dynamic ones. Static and dynamic attributes can be distinguished with scarce detailed knowledge of the event logs, and this distinction is commonly made in PPM [22]. First, some general design decisions are clarified, and then the implementation of further model parts is detailed. Fig. 2 provides a general overview of the architecture.

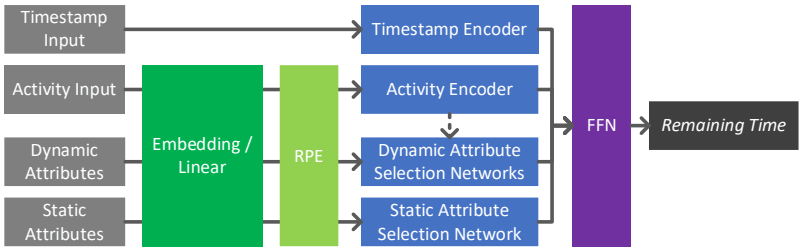

Fig. 2. Overview of the architecture of the developed transformer

4.1 Embedding and Positional Encoding

The input to the neural networks varies for the different types of attributes in the datasets and follows a general approach depending on the type, as described below.

Categorical Attributes. Nominal categorical attributes are encoded using learnable embeddings, as is common with transformers. This balances expressiveness and dimensionality and increases the model's capacity. A single embedding length is used to facilitate handling vector and matrix dimensions throughout the model.

Numerical Attributes. Min-max scaling is applied to all numerical attributes; thus, attributes are scaled linearly using the minimum and maximum derived from the training set. The same approach is used for ordinal categorical attributes, numbered according to their natural order in advance. As numerical cannot be embedded without discretization, a linear projection is applied to the scaled attributes to maintain identical dimensions for numerical and categorical attributes across the model.

Unlike RNNs, transformers do not impose positional information on their inputs, so positional encodings are usually added [5]. This work uses rotary position encoding (RPE), which combines relative and absolute position information and reportedly outperforms other encodings [25].

4.2 Activity and Timestamp Encoder

Activity and timestamps are special event log attributes, as both are central elements of an event log and are handled separately from other event log attributes. A transformer

encoder is used to encode both sequences for each of these attributes separately. The activity is embedded and positionally encoded beforehand, while the time features are handled as numerical attributes and concatenated.

Central to the transformer is the attention mechanism which is applied to the input sequences $X = \langle x_1, x_2, \ldots, x_n \rangle$ with x_i commonly being a vector representation of a sequence elements. In this case, X is an embedded sequence of activities. The input to the attention consists of queries in a matrix Q, keys of dimension d_k in a matrix K, and values in a matrix V [5]. In case identical sequences are used as input to the attention, such that $Q = XW_Q$, $K = XW_K$, and $V = XW_V$ with W denoting learned weight matrices, this is known as self-attention. Instances with different query and key matrices are commonly called cross-attention [6]. Multiple attentions are usually applied to the inputs separately and concatenated, leading to multi-head attention [5].

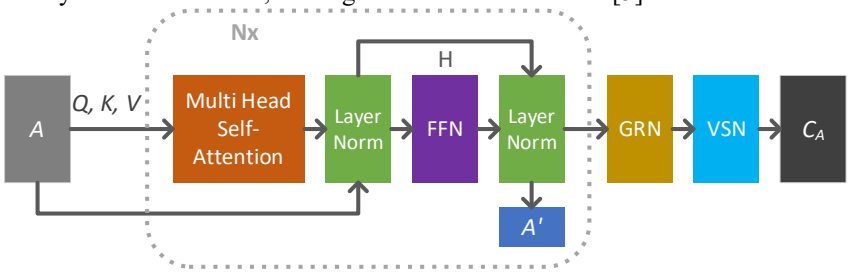

Fig. 3. Activity and timestamp encoder

The outputs of the multi-head attention are subsequently subjected to layer normalization and a position-wise feed-forward network. In addition, residual connections are added so that the encoded sequence X' as a result of self-attention on X with an activation function ϕ, which also describes a transformer encoder block, is given as:

$$
\begin{aligned}
H &= \text{LayerNorm}(\text{MultiHeadAttention}(Q, K, V) + X) \\
X' &= \text{LayerNorm}(\text{FFN}(H) + H) \\
\text{FFN}(a) &= \phi(W_1 a + b_1)W_2 + b_2 \\
\phi(x) &= \text{Mish}(x) = \tanh\big(\text{softplus}(x)\big)\, x
\end{aligned}
$$

The *Mish* [27] activation function, which has self-regularizing properties and is well-suited for deep networks, is used throughout the model. Multiple encoders are stacked four times, leading to a deeper model. The encoded activity and timestamp matrices are denoted as A' and T', respectively. The matrices are then flattened using a variable selection network [23], as shown in Fig. 3, similar to the dynamic attribute sequences.

4.3 Static Attribute Selection Network

A variable selection network (VSN) [23] is used to include the static attributes and handle them separately from the dynamic ones. VSNs are based on GRNs and provide a weighting of input attributes. Given several static attributes $S = [s_1, s_2, \ldots, s_m]$ with s_i being a preprocessed vector of an encoded attribute, a VSN weighs inputs by applying a GRN followed by a softmax function, resulting in a weight vector v_i and again feeding

each variable in its own GRN and summarizing the outputs. The final static context vector C_S is derived as follows:

$$C_S = \text{VSN}(S) = \sum_{i=1}^{m} v_i \, \tilde{s}_i$$
$$v = \text{softmax}\big(\text{GRN}(S)\big)$$
$$\tilde{s}_i = \text{GRN}_{s_i}(s_i)$$

The GLU and GRN are defined using the following equations and extensively use residual connections, with W indicating a trained weight matrix and a bias b. Please note that weights and biases are not shared between individual GLUs and GRNs.

$$\text{GRN}(a) = \text{LayerNorm}(a + \text{GLU}(W_1 \, \phi(W_2 \, a + b_2) + b_1)$$
$$\text{GLU}(a) = \phi(W_1 \, a + b_1) \circ (W_2 \, a + b_2)$$

4.4 Dynamic Attribute Selection Network

Given a sequence of dynamic attribute values, $D_k = \langle d_1^k, d_2^k, \ldots, d_n^k \rangle$ of the dynamic attribute k that is concurrent to the sequence of activities, each attribute is encoded separately applying self-attention first. The encoded dynamic attributes are then subjected to a cross-attention using the previously encoded activities. This setup is quite similar to the original transformer decoder [5]. However, no causal masking is applied. Please note that k is omitted in the following equation to improve clarity:

$$H_D = \text{LayerNorm}(\text{MultiHeadAttention}(Q_D, K_D, V_D) + D)$$
$$H'_D = \text{LayerNorm}(\text{MultiHeadAttention}(Q_{D'}, K_{A'}, V_{A'}) + H_D)$$
$$D' = \text{LayerNorm}(\text{FFN}(H'_D) + H'_D)$$
$$C_D = \text{VSN}(D')$$
$$Q_D = DW_Q^D, K_D = DW_K^D, V_D = DW_V^D \text{ and } Q_{D'} = D'_k W_Q^{D'}, K_{A'} = A'W_K^{A'}, V_{A'} = A'W_V^{A'}$$

This is done to extract importances from the attribute sequence itself and then exploit the relationship between the activities and further attributes, similar to [6]. After that, the encoded dynamic attribute sequence is flattened using a VSN to weigh the importance of the individual time steps, resulting in the attribute's dynamic context vector C_{Dk}, as shown in Fig. 4. While many solutions in PPM use simpler desequentialization mechanisms, such as global average pooling [10, 20] layers or simple flattening [6], we suspect that targeted feature selection might improve the model's prediction quality.

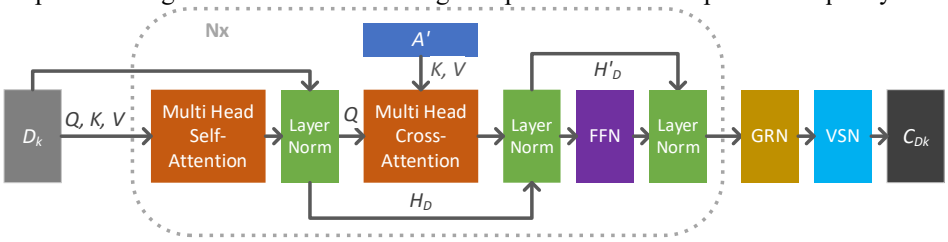

Fig. 4. Dynamic attribute selection network

5 Experimental Evaluation

Five datasets from different industries in ITSM's incident management are used to evaluate the developed architecture. These datasets show various event and case counts and event log attributes, ensuring a broad coverage and validity of the findings. The models are trained using early stopping up to a maximum of one hundred epochs on these logs.

- The "enriched event log of an incident management process" [28] derived from an IT company's ServiceNow platform and is referred to as "ServiceNow".
- The "Dataset belonging to the help desk log of an Italian Company" [29] originating in an Italian software company referred to as "Italian Company".
- The "Helpdesk" event log [30] from a software company without extra attributes.
- The BPIC 2014 [31] event log, created by Rabobank's IT organization.
- The BPIC 2013 [32] incident log from the car producer Volvo's IT organization.

5.1 Event Log Filtering and Splitting

The data undergoes a unified preprocessing to prepare it for the training. This involves removing duplicates and highly correlated attributes (≥ 0.9 or ≤ -0.9). All parts are combined using the provided identifiers for event logs with multiple files. The event logs are then split into 80/20 partitions using strict temporal splitting [8] to ensure unbiased and reproducible splits. Chronological outliers at the start and end of the BPIC 2014, BPIC 2013, and ServiceNow datasets are also filtered, and the 5% of cases with the most events are removed in all event logs as recommended [8].

Table 2. Event coverage of the event logs after filtering and strict temporal splitting [8]

	Train		Test		Total	
	Abs.	**Rel.**	**Abs.**	**Rel.**	**Abs.**	**Rel.**
Helpdesk	11,029	81.45%	2,512	18.55%	13,541	98.77%
Italian Company	15,807	77.08%	4,700	22.92%	20,507	96.06%
ServiceNow	96,350	77.69%	27,670	22.31%	124,020	87.52%
BPIC 2013	8,653	31.15%	19,124	68.85%	27,777	42.39%
BPIC 2014	318,228	80.13%	78,901	19.67%	397,129	85.09%

The strict temporal splitting approach performs well, capturing over 85% of events compared to unfiltered logs, as shown in Table 2. However, applying this method to the BPIC 2013 event log results in many invalid events between the train and test sets, rendering it unsuitable [8] for strict temporal splitting. Consequently, it was opted for a temporal split without debiasing for this event log. This leads to an acceptable coverage of 89.55% of the unfiltered event log, with 17.81% of events in the test set.

5.2 Evaluation Results

Predicting the remaining time is a regression task requiring metrics to determine the deviation between the label and the predicted values. Although the mean absolute error

(MAE) shows some sensitivity to outliers, it is less than other metrics and offers intuitive interpretability [33]. The mean squared error (MSE) complements this metric with a stronger penalization for larger errors and potential outliers [33]. To assess whether the models have learned adequately, a naïve model, which predicts only the train set's median for the remaining time, is added for reference. A baseline model is also implemented using an LSTM approach as described in [9], with the identical handling of numerical and categorical attributes as the transformer without positional encoding.

The evaluation results are shown in Table 3 and indicate that most models have learned better predictions than the naïve models. An exception to this is the Italian company event log, where the LSTM performs worse. This may indicate potential instability of the model that might be alleviated by using more attributes or different preprocessing in this model, especially using embeddings might not be suitable. Additional model fine-tuning might also be beneficial in the case of the LSTM.

Table 3. Results of the evaluation on the test set with the MAE and MSE in days

Event Log	Naïve		LSTM		Transformer		Mean	
	MAE	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**	**MAE**	**MSE**
Helpdesk	5.05	109.50	4.94	107.29	**3.55**	**68.71**	4.51	95.17
Italian Company	15.58	147.98	21.61	699.71	**7.81**	**113.97**	15.00	320.55
ServiceNow	3.26	31.68	2.29	**20.49**	**1.95**	22.66	2.50	24.94
BPIC 2013	4.26	28.88	2.95	14.21	**2.46**	**11.35**	3.22	18.15
BPIC 2014	1.93	14.37	1.79	**11.51**	**1.71**	13.22	1.81	13.03

The transformer model generally outperforms the LSTM and naïve models regarding MAE and MSE across most datasets. This suggests that it is a more accurate prediction model for the event logs used in this work, measured by MAE. Interestingly, despite the transformer model's significantly better MAE, the LSTM slightly outperforms the transformer in terms of MSE in some instances, suggesting a better handling of outliers.

Regarding the training, it must be stated that the deep transformer model exhibited significant overfitting in the first tries, while the LSTM was unproblematic in this regard. This necessitated a low learning rate and the extensive use of dropout, L2 regularization, and weight decay in the model. During training, linear warm-up epochs were incorporated at the start, and a linear decay rate was added for the later epochs. This was done to help the used adaptive AdamW optimizer learn accurate gradient statistics first and later decrease the magnitude of weight updates as the training progresses. This increased the epochs required to converge, again adding to the training time.

6 Discussion and Limitations

The empirical results in this work underscore the transformer's performance and feasibility in the novel architecture presented for the domain of ITSM. The developed architecture shows promise in handling ITSM processes' complexities and outperforms naïve baselines and LSTMs on several event logs. Notably, the utility of several architectural features new to PPM is shown, such as advanced positional encoding

techniques [25] and the distinguishing between static and dynamic attributes [22], which enable the model to effectively capture the inherent contextual and temporal dynamics of ITSM processes. As a novel approach, this complements earlier work that eschewed this distinction [6] or omitted additional attributes [10], offering insights into the design of transformer architectures. Integrating GLUs and GRNs also contributes to more refined handling of these attributes, replicating findings outside of PPM [23], showing their transferability, and improving the model's performance. Additionally, several insights into training transformer models could be gained, specifically regarding the mitigation of overfitting, which was found to be pivotal in such deep models.

Nonetheless, this study has some limitations. First, further ablation studies of the model might be necessary to assess the independent impact of each component, such as the VSNs and positional encoding. Also, as this work focused on a newly developed architecture, a comprehensive tuning of the hyperparameters was omitted. The evaluation results might be more favorable if separate tuning is applied for each event log. This should also be combined with a further comparison of more event logs and additional benchmark models to assess the architecture on a broader scale.

7 Conclusion and Future Research

This study introduced a novel transformer-based architecture for predictive process monitoring in IT Service Management (ITSM), addressing the challenges posed by the complexity and heterogeneity of ITSM processes. The proposed model incorporates advanced techniques such as rotary position encoding, GLUs, and separate handling of static and dynamic attributes, which have not been widely explored in predictive process monitoring. Generally, the transformer model demonstrates a robust capability to generalize across different datasets, as evidenced by its performance metrics attained in this work. The enhanced prediction capabilities of transformers in ITSM can drive operational effectiveness by facilitating decision-making and proactive SLA management. However, this work shows that deep transformers tended to overfit, necessitating careful regularization techniques during training. Addressing these challenges of overfitting and optimizing the training process will be crucial for advancing the application of these models in future works. Overall, the results validate the hypothesis that transformers can meet the complex demands of ITSM predictive process monitoring with appropriate modifications, potentially leading to more operational efficiency and service-level adherence. To ensure that optimal models are selected, future work on this architecture should include more extensive studies on separate model components of the transformer to assess their impact on prediction performance. This should be supplemented with an extended evaluation containing additional event logs and prediction targets, such as the next activity and timestamp.

Many relevant data sources in ITSM, such as configuration management databases and organizational and technical structures, form multi-layered network structures. In this regard, the work with non-tabular network data and geometric deep learning remains interesting for future analyses.

References

1. Bardhan, I.R., Demirkan, H., Kannan, P.K., Kauffman, R.J., Sougstad, R.: An Interdisciplinary Perspective on IT Services Management and Service Science. JMIS. 26, 13–64 (2010). https://doi.org/10.2753/mis0742-1222260402.
2. Rama-Maneiro, E., Vidal, J., Lama, M.: Deep Learning for Predictive Business Process Monitoring: Review and Benchmark. IEEE Trans. Serv. Comput. 16, 739–756 (2022). https://doi.org/10.1109/tsc.2021.3139807.
3. Loewenstern, D., Shwartz, L.: IT Service Management of Using Heterogeneous, Dynamically Alterable Configuration Item Lifecycles. In: Cordeiro, J. and Filipe, J. (eds.) 10th International Conference on Enterprise Information Systems. pp. 155–160. Barcelona, (2008).
4. Mao, H., Zhang, T., Tang, Q.: Research Framework for Determining How Artificial Intelligence Enables Information Technology Service Management for Business Model Resilience. Sustainability. 13, 11496 (2021). https://doi.org/10.3390/su132011496.
5. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is All You Need. In: Guyon, I., Luxburg, U. von, Bengio, S., Wallach, H.M., Fergus, R., Vishwanathan, S.V.N., and Garnett, R. (eds.) Advances in Neural Information Processing Systems. pp. 5998–6008. NeurIPS, Long Beach, CA, USA (2017).
6. Ni, W., Zhao, G., Liu, T., Zeng, Q., Xu, X.: Predictive Business Process Monitoring Approach Based on Hierarchical Transformer. Electronics. 12, 1273 (2023). https://doi.org/10.3390/electronics12061273.
7. Johannesson, P., Perjons, E.: An Introduction to Design Science. Springer, Cham, Switzerland (2021). https://doi.org/10.1007/978-3-030-78132-3.
8. Weytjens, H., De Weerdt, J.: Creating Unbiased Public Benchmark Datasets with Data Leakage Prevention for Predictive Process Monitoring. In: Marrella, A. and Weber, B. (eds.) Business Process Management Workshops. pp. 18–29. Springer, Cham, Switzerland (2022). https://doi.org/10.1007/978-3-030-94343-1_2.
9. Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive Business Process Monitoring with LSTM Neural Networks. In: Dubois, E. and Pohl, K. (eds.) Advanced Information Systems Engineering. pp. 477–492. Springer, Essen, Germany (2017). https://doi.org/10.1007/978-3-319-59536-8_30.
10. Bukhsh, Z.A., Saeed, A., Dijkman, R.M.: ProcessTransformer: Predictive Business Process Monitoring with Transformer Network, http://arxiv.org/abs/2104.00721, (2021).
11. 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. pp. 225–230. IEEE, Fukuoka, Japan (2020). https://doi.org/10.1109/icaiic48513.2020.9065057.
12. Chen, H., Fang, X., Fang, H.: Multi-task prediction method of business process based on BERT and Transfer Learning. Knowl. Based Syst. 254, 109603 (2022). https://doi.org/10.1016/j.knosys.2022.109603.
13. Moon, J., Park, G., Jeong, J.: POP-ON: Prediction of Process Using One-Way Language Model Based on NLP Approach. Appl. Sci. 11, 864 (2021). https://doi.org/10.3390/app11020864.
14. Wang, J., Yu, D., Liu, C., Sun, X.: Outcome-Oriented Predictive Process Monitoring with Attention-based Bidirectional LSTM Neural Networks. In: 13th International Conference on Web Services. pp. 360–367. IEEE, Milan, (2019). https://doi.org/10.1109/icws.2019.00065.
15. Jalayer, A., Kahani, M., Beheshti, A., Pourmasoumi, A., Motahari-Nezhad, H.R.: Attention Mechanism in Predictive Business Process Monitoring. In: 24th International Enterprise

Distributed Object Computing Conference. pp. 181–186. IEEE, Eindhoven, Netherlands (2020). https://doi.org/10.1109/edoc49727.2020.00030.
16.Jalayer, A., Kahani, M., Pourmasoumi, A., Beheshti, A.: HAM-Net: Predictive Business Process Monitoring with a hierarchical attention mechanism. Knowl. Based Syst. 236, 107722 (2022). https://doi.org/10.1016/j.knosys.2021.107722.
17.Rivera-Lazo, G., Ñanculef, R.: Multi-attribute Transformers for Sequence Prediction in Business Process Management. In: Pascal, P. and Ienco, D. (eds.) Discovery Science. pp. 184–194. Springer, Montpellier, (2022). https://doi.org/10.1007/978-3-031-18840-4_14.
18.Wickramanayake, B., He, Z., Ouyang, C., Moreira, C., Xu, Y., Sindhgatta, R.: Building Interpretable Models for Business Process Prediction using Shared and Specialised Attention Mechanisms. Knowl. Based Syst. 248, 108773 (2022). https://doi.org/10.1016/j.knosys.2022.108773.
19.Irwin, C., Dossena, M., Leonardi, G., Montani, S.: Structural Positional Encoding for Knowledge Integration in Transformer-based Medical Process Monitoring. In: Calimeri, F., Dragoni, M., and Stella, F. (eds.) 2nd AIxIA Workshop on Artificial Intelligence for Healthcare. pp. 18–30. CEUR, Rome, Italy (2023).
20.Wang, J., Huang, J., Ma, X., Li, Z., Wang, Y., Yu, D.: MTLFormer: Multi-Task Learning Guided Transformer Network for Business Process Prediction. IEEE Access. 11, 76722–76738 (2023). https://doi.org/10.1109/access.2023.3298305.
21.Amiri Elyasi, K., van der Aa, H., Stuckenschmidt, H.: PGTNet: A Process Graph Transformer Network for Remaining Time Prediction of Business Process Instances. In: Guizzardi, G., Santoro, F., Mouratidis, H., and Soffer, P. (eds.) Advanced Information Systems Engineering. pp. 124–140. Springer, Cham, Switzerland (2024). https://doi.org/10.1007/978-3-031-61057-8_8.
22.Brunk, J.: Structuring Business Process Context Information for Process Monitoring and Prediction. In: 22nd Conference on Business Informatics. pp. 39–48. IEEE, Antwerp, Belgium (2020). https://doi.org/10.1109/cbi49978.2020.00012.
23.Lim, B., Arik, S.O., Loeff, N., Pfister, T.: Temporal Fusion Transformers for Interpretable Multi-horizon Time Series Forecasting, http://arxiv.org/abs/1912.09363, (2020).
24.Wen, Q., Zhou, T., Zhang, C., Chen, W., Ma, Z., Yan, J., Sun, L.: Transformers in Time Series: A Survey. In: Elkind, E. (ed.) 32nd International Joint Conference on Artificial Intelligence. pp. 6778–6786. Macao, PRC (2023). https://doi.org/10.24963/ijcai.2023/759.
25.Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., Liu, Y.: RoFormer: Enhanced transformer with Rotary Position Embedding. Neurocomputing. 568, 127063 (2024). https://doi.org/10.1016/j.neucom.2023.127063.
26.Shazeer, N.: GLU Variants Improve Transformer, http://arxiv.org/abs/2002.05202, (2020).
27.Misra, D.: Mish: A Self Regularized Non-Monotonic Activation Function. In: 31st British Machine Vision Virtual Conference. Virtual (2020).
28.Amaral, C., Fantinato, M., Peres, S.: Incident management process enriched event log, https://archive.ics.uci.edu/dataset/498, (2018). https://doi.org/10.24432/c57s4h.
29.Polato, M.: Dataset belonging to the help desk log of an Italian Company, (2017). https://doi.org/10.4121/uuid:0c60edf1-6f83-4e75-9367-4c63b3e9d5bb.
30.Verenich, I.: Helpdesk, (2016). https://doi.org/10.17632/39bp3vv62t.1.
31.van Dongen, B.F.: BPI Challenge 2014, (2014). https://doi.org/10.4121/uuid:c3e5d162-0cfd-4bb0-bd82-af5268819c35.
32.Steeman, W.: BPI Challenge 2013, (2013). https://doi.org/10.4121/uuid:a7ce5c55-03a7-4583-b855-98b86e1a2b07.
33.Jadon, A., Patil, A., Jadon, S.: A Comprehensive Survey of Regression Based Loss Functions for Time Series Forecasting, http://arxiv.org/abs/2211.02989, (2022).