Using Heterogeneous Graph Neural Networks for Explainable Next-Activity Prediction

Vincenzo Pasquadibisceglie[1,2,*], Raffaele Scaringi[1,2], Annalisa Appice[1,2], Giovanna Castellano[1,2] and Donato Malerba[1,2]

[1]*University of Bari Aldo Moro, Bari, Italy*
[2]*Consorzio Interuniversitario Nazionale per l'Informatica - CINI, Bari, Italy*

Abstract

The prediction of the next-activity in the ongoing execution (event trace) of a business process provides useful information to take smart decisions for mitigating inefficiencies, preventing undesired activities and boosting business values of enterprises. In this paper, we describe PROPHET: a deep learning approach based on heterogeneous graph neural networks, to strike a balance between accurate predictions and interpretability in the next-activity prediction task. Specifically, it represents event traces as heterogeneous graphs by using different node types to express different characteristics of several events, and different types of graph links to express relationships between event characteristics, as well as relationships between events. It uses heterogeneous Graph Attention Networks (GATs), to achieve good accuracy in next-activity predictions. Finally, it integrates a modified version of the GNN Explainer algorithm to disclose explainable information related to characteristics, events and relationships between events that mainly influenced the prediction. Experiments with various benchmark event logs show the accuracy of PROPHET compared to several current state-of-the-art methods and draw insights from explanations recovered through the GNN Explainer algorithm.

Keywords

Next-activity prediction, Heterogeneous graph learning, Graph Neural Network, Multi-view Learning, eXplainable Artificial Intelligence

1. Introduction

Predictive Process Monitoring (PPM) is a family of predictive approaches to predict the unfolding of ongoing executions (event traces) of a business process based on knowledge learned from historical event logs. In particular, predicting the next-activity in an ongoing event trace is a crucial task in PPM as such predictive information may allow analysts to intervene proactively and prevent undesired behaviors. Notably, with the recent boom of deep learning, the PPM literature has achieved amazing results using different deep learning architectures, e.g., Long Short-Term Neural Network (LSTMs) [1, 2, 3], Convolutional Neural Networks (CNNs) [4, 5], Vision Transformers (ViTs) [6] and Large Language Models (LLMs) [7], to yield accurate predictions of the next-activity in ongoing event traces. On the other hand, multi-view learning has recently emerged as a fundamental deep learning direction for PPM approaches. In particular, in [1, 6], the multi-view learning strategy has allowed the development of flexible deep neural models that gain accuracy in next-activity prediction problems thanks to their ability to leverage the diversity of the information in the multiple characteristics of an event. Nevertheless, deep neural models generally learn opaque models, while easier-to-explain predictive models are becoming increasingly desirable in PPM applications [8, 2, 6]. Finally, in the last decade, graph representation learning has recently gained prominence across various domains, attracting intense research attention in deep learning [9]. Accordingly, a few recent PPM studies [10, 11, 12, 13] have explored the performance of various Graph Neural Network (GNN) architectures for graphs trained for PPM problems. However,

SEBD 2025: 33rd Symposium On Advanced Database Systems, June 16–19, 2025, Ischia, IT
[*]Corresponding author.
vincenzo.pasquadibisceglieds@uniba.it (V. Pasquadibisceglie); raffaele.scaringi@uniba.it (R. Scaringi); annalisa.appice@uniba.it (A. Appice); giovanna.castellano@uniba.it (G. Castellano); donato.malerba@uniba.it (D. Malerba)
0000-0002-7273-3882 (V. Pasquadibisceglie); 0000-0001-7512-7661 (R. Scaringi); 0000-0001-9840-844X (A. Appice); 0000-0002-6489-8628 (G. Castellano); 0000-0001-8432-4608 (D. Malerba)

previous PPM studies consider homogeneous graphs to mainly represent the activity control-flow and eventually use external data synopsis to encode the multi-view information associated with event nodes. In addition, the priority of previous graph-based PPM studies was mainly to provide accurate predictions of future states of ongoing event traces without paying extensive attention to explain predictions achieved with graph representations of event traces.

Based on the premises reported above, we describe a PPM graph-based approach, named PROPHET (explainable Predictive heteROgeneous graPH nEural neTworks), that has been recently presented in [14]. It trains a heterogeneous Graph Neural Network (GNN) model to predict the executing activity of the next event in an ongoing event trace and explains which characteristics of the ongoing trace mainly contribute to the decision. Specifically, we introduce the design of an event data engineering scheme to map ongoing event traces recorded in an historical event log into heterogeneous graphs by embracing the multi-view representation of event data. We illustrate a heterogeneous graph learning approach that integrates the attention mechanism implemented in the GAT layer [15] into a heterogeneous GNN. We describe an extension of the heterogeneous GNN Explainer algorithm [16], that is used to disclose how specific event and trace characteristics, as well as event relationships, influence the local decisions of the proposed predictive model. Finally, we illustrate the main results of an extensive experimentation, which show the capability of the presented approach to achieve accuracy comparable to related deep multi-view learning and graph learning approaches presented in the recent PPM literature, as well as disclose useful explanations of the model behavior in terms of the most informative inputs.

The paper is organized as follows. Section 2 reports the basics of the proposed approach. Section 3 describes the training stage of PROPHET, while Section 4 illustrate the experimental study to evaluate the accuracy and explainability performance of PROPHET. Finally, Section 5 draws conclusions.

2. Basics

An *event trace* is a record of a business process that shows the stages of its execution through a sequence of events. An *event* is portrayed from multiple views, where a *view* describes each event based on a specific characteristic. Specifically, each event is represented in two mandatory views associated with the activity A and the timestamp T (indicating when the activity occurred), respectively, and in m supplementary views, V_j with $j = 1, \dots, m$ associated with optional event characteristics (e.g., resource responsible for the activity or the cost involved in completing it). In addition, an event trace may be associated with n supplementary global characteristics, C_h with $h = 1, \dots, n$, which describe general trace information (e.g., the age of a patient at the time of her hospitalization). Let $\mathscr{A}$ be the set of activity names, $\mathscr{S}$ be the set of trace identifiers, $\mathscr{T}$ be the set of timestamps, $\mathscr{V}_j$ with $1 \leq j \leq m$ be the set of values in the j-th characteristic of an event, while $\mathscr{C}_h$ $1 \leq h \leq n$ be the set of values in the h-th global characteristic of an event trace. For simplicity, let us denote $\mathbf{C} = [C_1, \dots, C_n]$ – the vector of global trace characteristics , $\mathscr{C} = \mathscr{C}_1 \times, \dots, \times \mathscr{C}_n$ – the domain of $\mathbf{C}$, $\mathbf{V} = [A, T, V_1, \dots, V_m]$ – the vector of local characteristics and $\mathscr{V} = \mathscr{A} \times \mathscr{T} \times \mathscr{V}_1 \times \dots \times \mathscr{V}_m$ – the domain of $\mathbf{V}$.

Definition 1 (Event). *Given the event universe $\mathscr{E} = \mathscr{S} \times \mathscr{V}$, an event $e \in \mathscr{E}$ is a tuple $e = (s, a, t, v_1, \dots, v_m)$ that represents the occurrence of activity a in trace s at timestamp t with characteristics $v_1, \dots, v_m$.*

Let us introduce the functions $\pi_{\mathscr{S}} : \mathscr{E} \mapsto \mathscr{S}$ such that $\pi_{\mathscr{S}}(e) = s$, $\pi_{\mathscr{A}} : \mathscr{E} \mapsto \mathscr{A}$ such that $\pi_{\mathscr{A}}(e) = a$, $\pi_{\mathscr{T}} : \mathscr{E} \mapsto \mathscr{T}$ such that $\pi_{\mathscr{T}}(e) = t$, $\pi_{\mathscr{V}_j} : \mathscr{E} \mapsto \mathscr{V}_j$ such that $\pi_{\mathscr{V}_j}(e) = v_j$ and $j = 1, \dots, m$.

Definition 2 (Trace). *Let $\mathscr{E}^*$ denote the set of all possible sequences on $\mathscr{E}$. A trace is a pair $(\sigma, \mathbf{c}) \in \mathscr{E}^* \times \mathscr{C}$ where:*

1. *$\sigma = \langle e_1, \dots, e_t \rangle \in \mathscr{E}^*$ is an event sequence so that: (a) $\forall i = 1, \dots, t, \exists e_i \in \mathscr{E}$ such that $\pi_\sigma(\sigma, i) = e_i$ and $\pi_{\mathscr{S}}(e_i) = s$, and (b) $\forall i = 1, \dots, t-1, \pi_{\mathscr{T}}(e_i) \leq \pi_{\mathscr{T}}(e_{i+1})$.*
2. *$\mathbf{c} = (c_1, \dots, c_n) \in \mathscr{C}$ is the vector of the values of the optional global trace characteristics.*

Let us introduce the function $\pi_\sigma : \mathscr{E}^* \times \mathbb{N} \mapsto \mathscr{E}$ such that $\pi_\sigma(\sigma, i) = e_i$.

Definition 3 (Event log). *Given a business process , an event log $\mathscr{LOG} \in \mathscr{B}(\mathscr{E}^* \times \mathscr{C})$ is a collection of historical event traces recoded for a specific business process.*

Definition 4 (Prefix event sequence). *Given a sequence $\sigma \in \mathscr{E}^*$,* the prefix event sequence $\sigma^k = \langle e_1, \dots, e_k \rangle$ is the event sub-sequence of σ starting from the beginning of σ, with $1 \leq k = |\sigma^k| < |\sigma|$.

Notice that the activity $\pi_{\mathscr{A}}(e_{k+1}) = a_{k+1}$ corresponds to the next-activity of the prefix event sequence σ^k in σ, i.e., $next(\sigma^k) = \pi_{\mathscr{A}}(e_{k+1})$ with $e_{k+1} = \pi_\sigma(\sigma, k+1)$.

Definition 5 (Prefix trace). *Given the trace $(\sigma, \mathbf{c}) \in \mathscr{E}^* \times \mathscr{C}$, its prefix trace with length k is the pair $(\sigma^k, \mathbf{c})$ where σ^k is the the prefix event sub-sequence of σ with length equal to k, while $\mathbf{c}$ is the vector of global characteristic values recorded in the trace.*

A graph is a smart data structure that allows the representation of entity characteristics and entity relationships. In this study, we introduce a graph encoding strategy to transform a prefix trace into a heterogeneous graph.

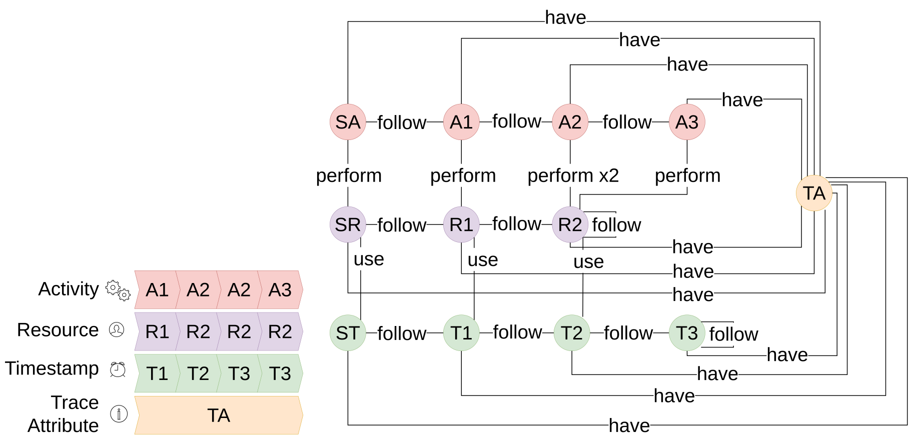
Definition 6 (Graph). *Let $\mathscr{G}$ denote the set of all graphs. A graph $g \in \mathscr{G}$ is an ordered pair $(\mathscr{N}, \mathscr{L})$, where $\mathscr{N}$ is the set of nodes, and $\mathscr{L}$ is the set of weighted links between nodes, i.e., $\mathscr{L} \subseteq \mathscr{N} \times \mathscr{N} \times \mathbb{R}$. Each node $n \in \mathscr{N}$ is associated with a node type and records a value. Each link $(n_1, n_2, w) \in \mathscr{L}$ is a link between nodes $n_1 \in \mathscr{N}$ and $n_2 \in \mathscr{N}$ associated with a numerical value $w \in \mathbb{R}$, called weight.*

Let $\mathscr{T}_{\mathscr{N}}$ be the set of distinct node types appearing in a graph g. In the *heterogeneous graph g* considered in this study $|\mathscr{T}_{\mathscr{N}}| > 1$. In this study, a prefix trace $(\sigma^k, \mathbf{c})$ is represented as an heterogeneous graph g that reserves a distinct type of nodes for each distinct type of characteristics. Hence, values of local event characteristics and global trace characteristics are recorded into $3 + m$ distinct types of nodes (i.e., $|\mathscr{T}_{\mathscr{N}}| = 3 + m$). In fact, a type of nodes is associated with the vector of global trace characteristics, a type of nodes is associated with the activity characteristic, a type of nodes is associated with the timestamp characteristic, and a type of nodes is associated with each one of the m optional event characteristics recorded in the event sequence of the prefix trace. Regarding links, g includes three types of links: (1) links pertaining intra-view relationships on the same local event characteristic observed in different events (i.e., links connecting two nodes belonging to the same node type); (2) links pertaining inter-view relationships involving two different local event characteristics measured in the same event (i.e., links connecting two nodes belonging to different node types); (3) links that establish a relationship between the vector of the global trace characteristics and each specific local event characteristic and (i.e., links connecting the node representing the global trace characteristics to any other node). Figure 1 shows an example of the transformation of a prefix trace into a heterogeneous graph.

Let $\mathscr{LOG} \in \mathscr{B}(\mathscr{E}^* \times \mathscr{C})$ be an event log, $\mathscr{P}_{\mathscr{G}} \subseteq \mathscr{B}(\mathscr{E}^* \times \mathscr{C} \times \mathscr{A})$ be the multiset of all heterogeneous graphs that represent prefix traces extracted from traces recorded in $\mathscr{LOG}$ and labeled with the next-activity values recorded in the corresponding traces. Let us consider a function $F\colon\ \mathscr{G} \mapsto \mathscr{A}$, such that $F(g)$ predicts the expected next-activity a_{k+1} of the graph g. Based on these premises, we frame the *next-activity prediction* task as a multi-class, heterogeneous graph classification problem. According to this formulation F can be trained by resorting to a graph-based learning technique from the multiset $\mathscr{P}_{\mathscr{G}}$ of the labeled heterogeneous graphs representing the prefix traces extracted from the event traces recorded in $\mathscr{LOG}$ and labeled with the next-activity in the corresponding traces.

3. The PROPHET approach

The PROPHET approach takes an event log LOG as input to learn the next-activity predictive function F. The learning stage is three-stepped.



(a) Prefix trace (b) Heterogenous graph

Figure 1: Example of conversion of a prefix trace into a heterogeneous graph.

In Step 1 PROPHET takes $\mathcal{LOG}$ as input and generates the multi-set of labeled prefix traces as output. To this purpose, it first transforms timestamp information recorded in an event into the time in seconds passed from the beginning of the trace. Then, it converts all numerical characteristics into categorical by applying the equal-frequency discretization algorithm as in [6]. The number of discretization bins of a characteristic is set equal to the average number of distinct categories in the original categorical characteristics recorded in $\mathcal{LOG}$. After this step, $\mathcal{LOG}$ contains all characteristics in the categorical format. Subsequently, it uses the Continuous-Bag-of-Words (CBOW) architecture of the Word2Vec scheme [17] to transform each categorical characteristic of a prefix trace extracted from $\mathcal{LOG}$ into a numeric representation. CBOW is an embedding approach that was originally formulated in the field of natural language processing and as recently used in several PPM approaches (e.g., [3, 6]). It uses a feed-forward neural network to predict a target value from the neighbored context. In this work, a distinct CBOW architecture is trained for each characteristic recorded in the prefix traces.

In Step 2 PROPHET converts each labeled prefix trace $(\sigma^k, \mathbf{c}, a_{k+1})$ into a labeled heterogeneous graph (g, a_{k+1}). For each event $e \in \sigma^k$, every event characteristic recorded in e is mapped into a graph node that contains the embedding representation of the characteristic value The embedding is the one computed with the corresponding CBOW model. The type of the node is that one-to-one associated with the mapped characteristic. At the same time, the embeddings of the values the global trace characteristics collected in $\mathbf{c}$ are recorded into the trace-based node of the graph. Once the nodes of g have been generated, the links between the created nodes are produced. Regarding inter-view links (which pertain nodes of different types), in this work, we generate the inter-view links considering the graph nodes whose types are associated with each pair of consecutive views in the vector $\mathbf{V}$ that represent the event characteristics recorded in $\mathcal{LOG}$. Let $\mathcal{P}_{\mathcal{G}}$ be the the multi-set of labeled heterogeneous graphs extracted from $\mathcal{LOG}$.

In Step 3 PROPHET learns a GNN model equipped with L of GAT layers [15] to obtain a next-activity prediction function F from $\mathcal{P}_{\mathcal{G}}$. The GAT learns iteratively new node representations, differentiating the learning process concerning the link types that are represented in the graph, and aggregating information coming from each neighborhood.

Finally, the GAT model is equipped with a variant of GNN Explainer [16] that allows us to explain the decisions of the GAT model by extracting insights on how each node and link contribute to a decision. The basic implementation of the GNN Explainer algorithm works as follows. Given a heterogeneous graph $g = (\mathcal{N}, \mathcal{L})$, the prediction $\tilde{y} = F(g)$ can be equally written as $\tilde{y} = F(\mathbf{X}, \mathbf{A})$, where $\mathbf{X}$ and $\mathbf{A}$ are the list of data feature matrices and the list of adjacency matrices that are, respectively, associated with g. The basic GNN Explainer algorithm explains this decision by jointly computing the following outputs:

a node feature mask $\hat{\mathbf{X}} = [\hat{\mathbf{X}}_{\mathbf{t_n}} \in [0,1]^{1\times d_{t_n}}, \forall t_n \in \mathcal{T}_{\mathcal{N}}]$ and a link mask $\hat{\mathbf{A}} = [\hat{\mathbf{A}}_{\mathbf{t_l}} \in [0,1]^{|\mathcal{N}_{t_a}|\times|\mathcal{N}_{t_b}|}, t_l = (t_a, t_b), \forall t_l \in \mathcal{T}_{\mathcal{L}}]$. The algorithm attempts to perturb the input graph g, composing a new "masked" graph $\hat{g} = (\hat{\mathcal{N}}, \hat{\mathcal{L}})$ so that $\mathbf{X}^m = [{\mathbf{X}_{\mathbf{t_n}}}^m = {\mathbf{X}_{\mathbf{t_n}}}^m \odot \hat{\mathbf{X}}_{\mathbf{t_n}}, \forall t_n \in \mathcal{T}_{\mathcal{N}}]$ is the list of data feature matrices associated with $\hat{g}$. The higher the value of $\hat{\mathbf{X}}_{\mathbf{t_n}}[i]$, the higher the importance of the corresponding i-th piece of information on the prediction. Notably, $\hat{\mathbf{X}}_{\mathbf{t_n}}[i]$ is a unique value estimated within for the nodes of g with type t_n. On the other hand, the higher the value of $\hat{A}_{t_l}[i,j]$, the more important the corresponding link. So, the main issue of this basic algorithm is that it cannot identify the most important node subset in the graph. For this reason, we adapted GNNExplainer to estimate the most important subgraph for a given prediction. Specifically, we modified the shape of parameter $\hat{\mathbf{X}}_{\mathbf{t_n}} \in [0,1]^{|\mathcal{N}_{t_n}|\times 1}, \forall t_n \in \mathcal{T}_{\mathcal{N}}$ keeping the rest of the algorithm unchanged comprising the element-wise multiplications to obtain the masked graph $\hat{g}$. In this way, the higher the value estimated for $\hat{\mathbf{X}}_{\mathbf{t_n}}[i]$ the more important the effect of the i node on the decision.

4. Experimental study

In this section, we describe some results of the experimentation conducted for evaluating the accuracy and explainability performance of PROPHET.

4.1. Event logs and experimental set-up

Table 1
Event log description: **domain** (F– Finance, M – Maintenance, A – Administration, C – Customer Service, H – Healthcare) of the business process, number of traces (#**Trace**), number of events (#**Event**), number of activities (#**Activities**), mean length of traces (**Avg length**), number of global trace characteristics (# **Trace characteristics**) and number of local event characteristics (# **Event characteristics**)

Log	Domain	#Trace	#Events	#Activities	Avg length	#Trace characteristics	#Event characteristics
BPI12AC	F	13087	60849	10	5	1	3
BPI12WC	F	9658	72413	6	7	1	3
BPI13O	M	819	2351	5	3	3	6
BPI17O	F	42995	193849	8	5	5	4
BPI20R	A	6886	36796	19	5	3	4
Helpdesk	C	3804	13710	9	4	0	2
Invoice	F	5123	62740	25	12	5	3
Logboek	H	1897	6973	10	4	4	3
SP2020	C	23906	178078	13	7	3	2

We used nine real-life event logs that are all available on the 4TU Centre for Research [1], except for SP2020 that is available on Zenodo[2]. A summary of the characteristics of the event logs considered in the evaluation is reported in Table 1. For each event log, a temporal split was done to divide the log into training and testing traces. Specifically, we sorted the event traces of each event log by their starting timestamps. The first two-thirds of the sorted traces were chosen for training the predictive model, while the remaining one-third was reserved for evaluating the model's performance on unseen data. The implementation of PROPHET is available online. [3]

4.2. Accuracy performance analysis

We evaluated the accuracy performance of both PROPHET and several state-of-the art deep neural methods formulated in the PPM field for the next-activity prediction task. The related methods considered in the evaluation study train: a LSTM model in [1], a BiLSTM model in [18], a Transformer model in [19], a ViT model [6], a CNN with Inception model in [4], a GCN model in [12] and GRNN

[1]https://data.4tu.nl/portal
[2]https://zenodo.org/records/3928487
[3]https://github.com/vinspdb/PROPHET

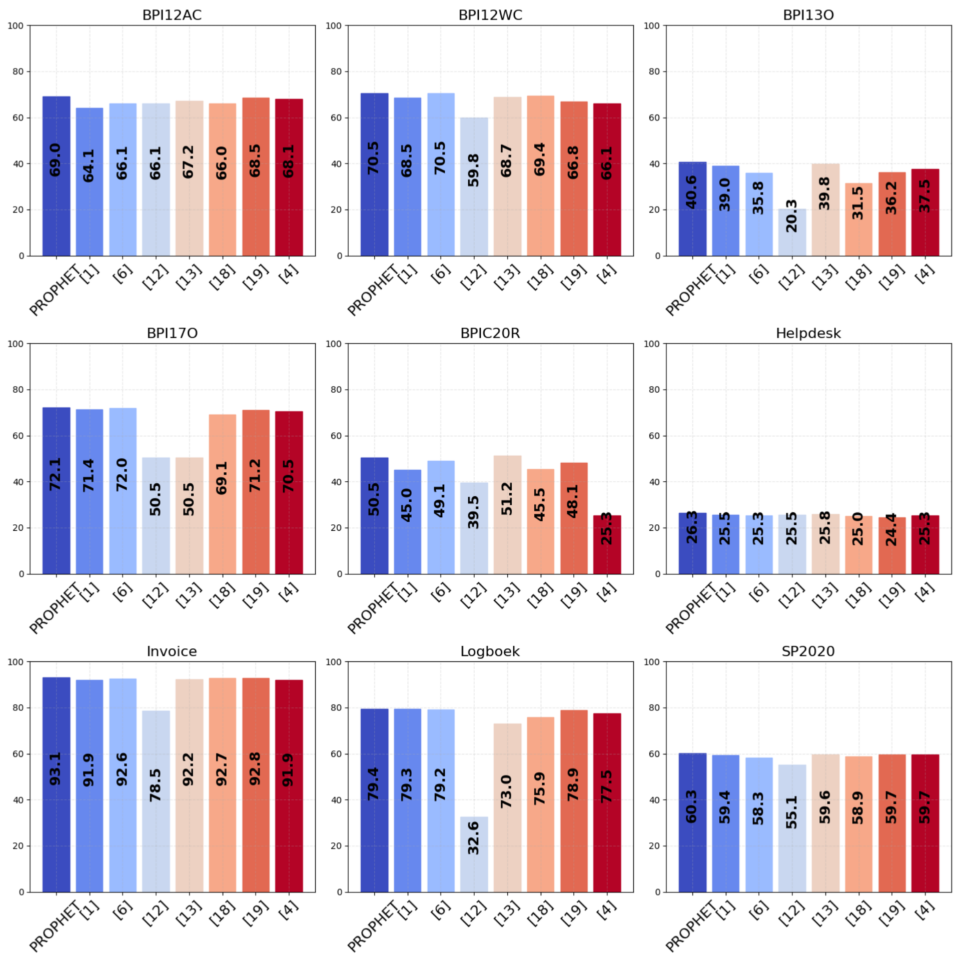
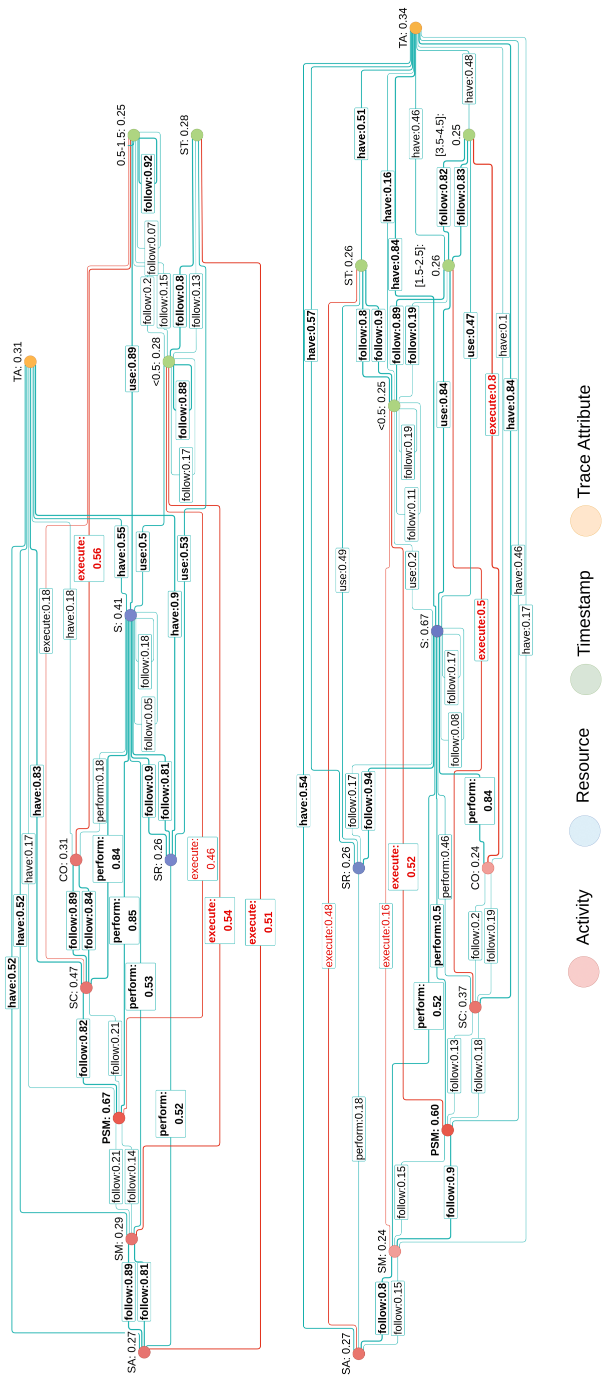


Figure 2: Macro FScore of PROPHET and the related deep learning methods [1], [6], [12], [13], [18], [19], [4]

model concatenated with a LSTM model in [13]. The methods described in [1], [18] and [19] adopt a sequence representation of prefix traces. The methods described in [6] and [4] adopt a color image representation of prefix traces. The method described in [12] adopts homogeneous instance graphs to represent prefix traces, while the method described in [13] handles homogeneous graphs extracted by replaying prefix traces on a Petri Net discovered from the event log and event-related temporal data extracted from the remaining views recorded in the event log. All related methods, except for [18] and [19], were originally formulated within the multi-view learning schema. However, to provide a fair comparison, we ran all related methods by accounting for all views recorded in the considered event logs. For each event log, for each PPM method, we measured the accuracy of each next-activity predictive function through the macro FScore. computed as $FScore = \frac{1}{k}\sum_{i=1}^{k} FScore_i$, where k is the number of distinct activities recorded in the event log. This multi-class accuracy metric was selected as it is commonly measured in imbalanced domains. Figure 2 shows the macro FScore measured for both PROPHET and the related methods. These results show that PROPHET commonly outperforms the related methods having [13] as runner-up. In particular, the method described in [13] performs better than PROPHET in BPIC20R only, by training a GRNN model followed by an LSTM model. In any case, PROPHET is the runner-up of the comparative analysis in this event log also.

4.3. Explanation performance analysis

In this section, we illustrate some examples of the decision explanations produced by PROPHET. Figures 3a and 3b show the heterogeneous graphs of two prefix traces recorded in the testing set of the event log



(a) "ME" (b) "CS"

Figure 3: Explanation of the next-activity decision of PROPHET (with next-activities: "ME" – "Manual enter the order number-Entered"; "CS" – "Compare of Sums missing") for two prefix traces recorded in the testing log of Invoice. The real values reported on nodes and links quantify the effect of the corresponding nodes or links on the decision. Values greater than 0.5 are in bold, while the links "execute" connecting activities and timestamps are colored in red.

Invoice and enriched with the explanations obtained with the extension of GNN Explainer algorithm. PROPHET correctly predicts "ME" – "Manual enter the order number-Entered" and "CS" – "Compare of Sums missing" as the next activity of the two ongoing traces, respectively. Both prefix traces share the same "SM - PSM - SC - CO" activity prefix, where: "SM" – "Start Missing", "PSM" – "Process Start Missing", "SC" – "Status Change to being approved missing" and "CO" – "Check Order number missing".

However, the two prefix traces represent ongoing executions of the Invoice business process, which will proceed with executing two different activities in the future. Therefore, this example shows a case where the control flow of activities does not provide enough information to correctly predict the two different next activities for the two prefix traces. Instead, PROPHET is able to yield the correct next-activity decision for both prefix traces by leveraging the multi-view information recorded in the event log. In addition, the explanation of the decisions highlights which characteristics already recorded for the ongoing trace helps PROPHET model to disentangle the next-activity ”ME” from the next-activity ”CS” in the considered prefix traces. In particular, the produced explanations show that, although the activities executed in the two prefix traces are all performed by the resource ”S” (Server), some differences occur in their time cycle by highlighting which information of each time cycle is actually relevant for predicting the correct next activity in the two cases. Notably, activities ”SM” and ”PSM” have been executed before 0.5 seconds passed from the beginning of the trace. However, in Figure 3a, activities ”SC” and ”CO” have been executed in the time interval that goes from 0.5 to 1.5 seconds passed from the beginning of the trace with the timestamp of ”CO” that has high relevance (greater than 0.5) on the correct decision ”ME” as the next-activity of the prefix trace. In Figure 3b, activities ”SC” and ”CO” have been executed in the time intervals that go from 1.5 to 2.5 seconds and 2.5 to 4.5 seconds passed, respectively, from the beginning of the trace. Notably, the timestamps of ”PSM”, ”SC” and ”CO” have all high relevance (greater than 0.5) on the correct decision ”CS” as the next activity of the prefix trace. Hence, the decision is explained as follows: the longer the time passed from the beginning of the ongoing trace before the execution of activities ”SC” and ”CO”, the more important the symptoms indicating ”CS” as the upcoming activity of the ongoing trace. This information can be used at the process management level to plan the work requested to better manage the execution, as well as the consequences of the execution of activity ”CS” on the current process execution.

5. Conclusion

In this work, we present a PPM approach that represents ongoing traces of a business process as heterogeneous graph and learns GAT model for the next-activity prediction task. In addition, it resorts to an extended version of the GNN Explainer algorithm to equip GAT’s decisions with explanations regarding trace characteristics mainly affecting the decision An extensive experimentation shows the accuracy of the proposed approach also compared to several related PPM methods. In addition, it provides some examples of the produced explanations. As future work, we plan to explore how concept drift detection may be integrated in the proposed graph-based approach, in order to keep the model accurate over time. In addition, we intend to start the investigation of how next-activity decision explanations disclosed with the proposed approach may be leveraged in a prescriptive PPM perspective.

Declaration on Generative AI

The authors have not employed any Generative AI tools.

Acknowledgments

Vincenzo Pasquadibisceglie and Giovanna Castellano are partially supported by the project FAIR - Future AI Research (PE00000013), Spoke 6 - Symbiotic AI, under the NRRP MUR program funded by the NextGenerationEU. Raffaele Scaringi is funded by a Ph.D. fellowship within the framework of the Italian “D.M. n. 352, April 9, 2022” - under the NRRP, co-supported by “Exprivia S.p.A.” (CUP H91I22000410007). The work of Donato Malerba is in partial fulfillment of the project MAD - The Additive Metamorphosis of Design (ARS01_00717) funded by MUR.

References

[1] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, A multi-view deep learning approach for predictive business process monitoring, IEEE Trans. Serv. Comput. 15 (2022) 2382–2395. doi:10.1109/TSC.2021.3051771.

[2] L. Aversano, M. L. Bernardi, M. Cimitile, M. Iammarino, C. Verdone, A data-aware explainable deep learning approach for next activity prediction, Eng. App. of Artif. Intell. 126 (2023) 106758. doi:https://doi.org/10.1016/j.engappai.2023.106758.

[3] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, DARWIN : An online deep learning approach to handle concept drifts in predictive process monitoring, Eng. Appl. Artif. Intell. 123 (2023) 106461. doi:10.1016/J.ENGAPPAI.2023.106461.

[4] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, Predictive process mining meets computer vision, in: BPM Forum 2020, volume 392 of *LNBIP*, Springer, 2020, pp. 176–192.

[5] H. Weytjens, J. De Weerdt, Process outcome prediction: Cnn vs. lstm (with attention), in: BPM 2020 International Workshops, Springer, 2020, pp. 321–333.

[6] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, JARVIS: Joining adversarial training with vision transformers in next-activity prediction, IEEE Trans. Serv. Comput. (2023) 1–14. doi:10.1109/TSC.2023.3331020.

[7] V. Pasquadibisceglie, A. Appice, D. Malerba, LUPIN: A LLM approach for activity suffix prediction in business process event logs, in: 6th International Conference on Process Mining, ICPM 2024, IEEE, 2024, pp. 1–8. doi:10.1109/ICPM63005.2024.10680620.

[8] R. Galanti, M. de Leoni, M. Monaro, N. Navarin, A. Marazzi, B. Di Stasi, S. Maldera, An explainable decision support system for predictive process analytics, Eng. App. of Artif. Intell. 120 (2023) 105904. doi:https://doi.org/10.1016/j.engappai.2023.105904.

[9] M. Li, A. Micheli, Y. G. Wang, S. Pan, P. Lió, G. S. Gnecco, M. Sanguineti, Guest editorial: Deep neural networks for graphs: Theory, models, algorithms, and applications, IEEE Trans. Neural Networks Learn. Syst. 35 (2024) 4367–4372. doi:10.1109/TNNLS.2024.3371592.

[10] S. Weinzierl, Exploring gated graph sequence neural networks for predicting next process activities, in: BPM 2022 International Workshops, Springer, 2022, pp. 30–42.

[11] I. Venugopal, J. Töllich, M. Fairbank, A. Scherp, A comparison of deep-learning methods for analysing and predicting business processes, in: IJCNN 2021, IEEE, 2021, pp. 1–8. doi:10.1109/IJCNN52387.2021.9533742.

[12] A. Chiorrini, C. Diamantini, L. Genga, D. Potena, Multi-perspective enriched instance graphs for next activity prediction through graph neural network, J. Intell. Inf. Syst. (2023) 1–21.

[13] E. Rama-Maneiro, J. C. Vidal, M. Lama, Embedding graph convolutional networks in recurrent neural networks for predictive monitoring, IEEE Trans. Knowl. Data Eng. 36 (2024) 137–151. doi:10.1109/TKDE.2023.3286017.

[14] V. Pasquadibisceglie, R. Scaringi, A. Appice, G. Castellano, D. Malerba, PROPHET: Explainable predictive process monitoring with heterogeneous graph neural networks, IEEE Trans. Serv. Comput. 17 (2024) 4111–4124. doi:10.1109/TSC.2024.3463487.

[15] S. Brody, U. Alon, E. Yahav, How attentive are graph attention networks?, in: ICLR 2022, 2022, pp. 1–26.

[16] Z. Ying, D. Bourgeois, J. You, M. Zitnik, J. Leskovec, GNNExplainer: Generating explanations for graph neural networks, in: NeurIPS 2019, 2019, pp. 9240–9251.

[17] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: ICLR 2013, 2013.

[18] B. Wickramanayake, Z. He, C. Ouyang, C. Moreira, Y. Xu, R. Sindhgatta, Building interpretable models for business process prediction using shared and specialised attention mechanisms, Knowledge-Based Systems 248 (2022) 1–22. doi:https://doi.org/10.1016/j.knosys.2022.108773.

[19] Z. A. Bukhsh, A. Saeed, R. M. Dijkman, Processtransformer: Predictive business process monitoring with transformer network, CoRR abs/2104.00721 (2021).