2024 6th International Conference on Process Mining (ICPM) | 979-8-3503-6503-0/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICPM63005.2024.10680620

LUPIN: A LLM Approach for Activity Suffix Prediction in Business Process Event Logs

1st Vincenzo Pasquadibisceglie
Department of Computer Science
University of Bari Aldo Moro
Bari, Italy
vincenzo.pasquadibisceglie@uniba.it

2nd Annalisa Appice
Department of Computer Science
University of Bari Aldo Moro
Bari, Italy
annalisa.appice@uniba.it

3rd Donato Malerba
Department of Computer Science
University of Bari Aldo Moro
Bari, Italy
donato.malerba@uniba.it

Abstract—**Forecasting future states of running process instances is one of the main challenges of Predictive Process Monitoring (PPM). Several deep learning approaches have recently achieved a valuable accuracy performance by addressing this task. On the other hand, with the recent boom of Large Language Models (LLMs) in multiple fields, LLMs have started attracting attention in PPM research also. In this study, we leverage the rich context of textual data to transform information recorded in event logs in smart textual data ready for boosting accurate PPM learning. In detail, we propose LUPIN, a LLM approach to predict the activity suffix of a running process instance. First it encodes historical running process instances in semantic text stories formulated according to narrative templates that account for information recorded in the event log. Then it fine tunes a pre-trained LLM model – medium BERT – on the text stories of historic running instances of a business process, to predict the activity suffix of any future running instance of the same business process. Finally, LUPIN integrates the XAI Integrated Gradient (IG) algorithm to explain how each part of the textual description of a running process instance has an effect on the prediction of its activity completion. The experimental evaluation explores the accuracy performance of LUPIN compared to that of several related methods and draws insights from the explanation retrieved through the IG algorithm.**

Index Terms—**PPM, LLMs, BERT, XAI, Activity Suffix Prediction, Fine Tuning**

I. Introduction

Predictive Process Monitoring (PPM) plays a crucial role in the modern business process management as it produces predictions of future states of a process execution given its running instance (i.e., trace of events) and the history of the past observed instances [1]. The main PPM objectives include predicting the remaining execution time [2], the next-activity [3], [4], or the outcome [5], [6] of a running process instance.

Over the past five years, the PPM literature has conferred an increasingly important role to deep learning as a powerful Artificial Intelligence (AI) paradigm to handle complex process behaviors and data structures within event logs, in order to learn useful knowledge to perform accurate predictions of the expected behaviour of a process instance. The recent PPM literature has explored the performance of various deep neural network architectures, including LSTMs [7], [8], CNNs [9], [10], GANs [11] in various PPM tasks. On the other hand, due to the recent notable focus on Large Language Models (LLMs), mainly exemplified by the use of Generative Pretrained Transformers (GPTs), the process mining community has started realizing the advantage of using LLMs in process mining, particularly, in PPM problems. LLMs allow us to capture complex patterns, context, and semantics in natural language. As they are pre-trained on rich linguistic data, they can be used in various predictive tasks without the need for extensive task-specific training datasets [12]. In addition, thanks to their ability to understand prompts and generate human-like text, LLMs can simplify the interaction of human stakeholders with PPM algorithms paving the way for a symbiotic PPM paradigm. However, as discussed in [12], the effective adaptation of LLMs to process mining and, consequently, to PPM tasks remains an open challenge. Initial studies exploring LLMs in tasks of next-activity, activity suffix or outcome prediction are described in [13]–[15]. In this paper, we focus on the PPM task of the activity suffix prediction that aims at predicting the sequence of activities until the completion of a running process instance.

We propose LUPIN: a **L**LM approach for s**U**ffix **P**redict**I**o**N** in business process event logs. Boosted by our recent results [3], [4] achieved with the next-activity prediction task, we adopt a multi-view learning approach. This consists in learning with multiple views of input event information (e.g., activity, resource, timestamp, cost) recorded in an event log to improve the generalization performance of the predictive model. In addition, inspired by the research results recently achieved from [13] in the context of the next-activity prediction, we convert the running process instances recorded in an event log into the form of semantic textual stories by using natural language-based utterances to add a semantic context to event information. Although the semantic context adds complexity to the event log representation, it also adds meaning to the process instance stories. Our work, similarly to that of [13], is founded on the hypothesis that accounting for a semantic context of the event data observed in a process instance can allow us to gain accuracy with the PPM model development. In particular, the stories are produced according to a narrative template that accounts for the multiple views recorded in the event log. However, differently from [13], where the narrative considers the sequence of activities recorded in the running instance enriched with the multi-view information recorded in the last event of the running process instance, we consider the multi-view information recorded in the entire "sequence of events" of a running instance. With regard to the event log representation, this study also advances [16] that is the seminal PPM study exploring the use of a LLM-based approach for activity suffix prediction. In fact, the authors of [16] handle activity and timestamp neglecting information recorded in the event log in any other view. So, the design of a LLM approach, which can actually handle the multi-view information recorded in each event of the entire process instance, is a practical contribution of this study, which represents an advancement with respect to the previous LLM-based PPM studies that commonly neglect multiple information recorded in all events.

In this study, we use the bi-directional LLM model – medium BERT, that is a compact pre-trained BERT variant recently introduced in [17]. This is selected among the plethora of pre-trained LLMs described in [17] as it is a compact LLM that achieves high accuracy despite it is developed with small computation resources.[1] We integrate medium BERT as part of the multi-output deep neural network that LUPIN uses to perform the activity suffix prediction of a running instance. The multi-output is used to map the consecutive activities of a suffix as the multiple targets of the deep neural model. Hence, the LUPIN network is composed of the pre-trained medium BERT whose output is fed into the multi-output classification heads. The entire network is fine tuned on the running instances observed for any specific business process before being used for the suffix prediction. Hence, LUPIN differs from traditional suffix prediction methods, which mainly rely on next-activity prediction models to complete a running process instance by repeatedly predicting the next-activity until the end of the trace has been reached. Instead, we predict "all" upcoming activities in a "single" prediction step.

Another key feature of LUPIN is the explainability of predictions achieved with the Integrated Gradients (IG) algorithm [18]. This is an eXplainable AI algorithm that scores the word tokens in the text story produced to describe a running process instance. These scores rank the part of the text story and highlight the process instance characteristics that most influenced the activity suffix prediction. This focus on the explainability of PPM decisions achieved with a LLM is a practical contribution of this study compared to existing LLM studies in PPM [13], [16]. To the best of our knowledge, previous studies do not consider yielding explanations for LLM decisions. On the other hand, the explainable topic has recently gained great attention in PPM, despite it has been mainly investigated with non-LLM PPM approaches, e.g., [4].

In short, the main contributions of this paper are listed in the followings. (1) We formulate a novel multi-view learning approach for activity suffix prediction, which uses text engineering to represent running process instances as text stories. (2) We fine tune a multi-output deep neural network that integrates medium BERT to the text stories of a business process history, in order to perform the activity suffix prediction for new running instances of the same business process. (3) We perform an experimental study of the accuracy of the proposed approach for activity suffix prediction in six event logs. This evaluation shows that handling context-aware inputs allows us to gain accuracy with the PPM model development compared to several PPM approaches from the recent PPM literature for the activity suffix prediction, which just use the event log for training. (4) We analyse some explanation achievements obtained with IG, to identify which pieces of the running instance mainly determine the prediction of the activity suffix.

The paper is organized as follows. Section II overviews recent advances of literature in the activity suffix prediction task. Section III describes the proposed LUPIN approach, while Section IV presents the event logs considered for the empirical evaluation, describes the experimental setting and illustrates relevant results achieved in the evaluation. Finally, Section V draws conclusions and sketches the future work.

II. Related Works

In the recent PPM literature, several deep learning approaches have been proposed to predict the activity suffix of a running instance. The majority of the literature approaches adopts an LSTM neural network trained to solve the problem of the next-activity prediction by accounting for activity, timestamp and, in a few cases, resource data. In particular, [7] adopts the one-hot representation of the activity coupled with multiple time-based features extracted from the events to encode a running process instance. First it trains a multi-output LSTM neural network to predict both the next-activity and the timestamp of the next-activity of a running instance. Then it uses the information predicted for the next-event to complete the instance and proceed in the suffix prediction by repeating the prediction until achieving the completion of the instance. [19] is a variant of [7] that uses an embedding of the sequence of activities to avoid the sparse matrix issue caused by one-hot encoding in presence of a high number of distinct activities. However, this approach accounts for activity information only. Instead, [8] uses a multi-output that considers the activity, timestamp and role of the resource to represent running instances. Similarly to [19], it learns an embedding representation of both activity and resource information and applies iteratively the multi-output LSTM neural network to repeatedly predict the next-activity, next-role and time of the next-activity until the suffix completion.

The approaches described above re-use a (multi-output) deep neural network already trained to predict the next-event to predict the activity completion of the running instance also. They complete the prediction of the instance suffix by applying repeatedly the one-step predictive model to the sequence of observed and predicted events. However, this mechanism may incur in the propagation of the prediction error across multiple steps whenever wrong predictions are used as part to the prefix processed to complete the prediction of the suffix. The issue of the error propagation for multi-step forecasting has been mainly explored in time series analysis [20]. This phenomenon is accentuated by increasing the number of

[1]We note that any LLM can be tested in the proposed approach.

multi-view characteristics predicted [7], [8]. Based on these premises, [11] has recently introduced a different mechanism to complete the suffix prediction. This study describes an encoder-decoder LSTM neural network trained as a GAN to predict the activity suffix in a single step. The encoder-decoder generates activity suffixes to mislead the discriminator, that is a LSTM distinguishing fake suffixes generated by the generator from real suffixes. This approach uses activity and time information. Following the same research direction, [21] proposes an attention-based encoder-decoder neural network that is trained to predict the activity suffix in a single step. The encoder considers different information like activities, multiple time-based features, and resources, whereas the decoder predicts the activities of the suffix only.

The approaches described in [11] and [21] are the closer to the approach illustrated in this paper, since all learn a deep neural model to predict the entire activity suffix in a single step. In fact, we adopt a multi-output learning approach for predicting the entire activity suffix in a single step. Notably, the multi-output learning paradigm is a well assessed learning paradigm to handle the possible interactions between multiple outputs simultaneously [22]. This ability of the multi-output learning paradigm offers the opportunity to handle interaction between activities of a process instance suffix without having to consider wrong predictions of future activities that are still unobserved, as it may happen with repeatedly applying one-step prediction. However, the related approaches consider activity, timestamp and, eventually, resource information only, while an event log can record several optional, process-dependent characteristics, such as the role of the resource or the cost of performing the activity. Recent PPM studies [3], [4] have shown that the richer and the more varied the processed event log information, the higher the learning ability of the PPM model. So, according to these conclusions, we formulate a LLM approach to account for information recorded in any view of an event log and represent this information according to a narrative template that is defined based on the views recorded in the event log. As an additional difference, related approaches train a deep neural model from scratch on the event logs, while we use a powerful, pre-trained LLM model that can be fine tuned on each new event log.

Regarding the use of LLMs in PPM, the idea of using an LLM model for activity suffix prediction has been introduced in [16] that compares the performance of several deep learning architectures designed for activity suffix prediction. However, this literature study considers activities and timestamps only by handling the the one-hot encoding of the activity and the timestamp to produce a text description. In addition the text description of a running process instance is a list of numeric values produced neglecting the context defined with a narrative template, as we have done in this study. As an additional difference, [16] considers the architectures of BERT and GPT trained from scratch without leveraging the pre-trained, foundation models trained with big linguistic resources.

Finally, none of the related approaches reported above explain decisions produced as activity suffix predictions, while we describe an approach that integrates the IG algorithm [18] to identify the most important contributing parts of the story of a running instance for the prediction produced.

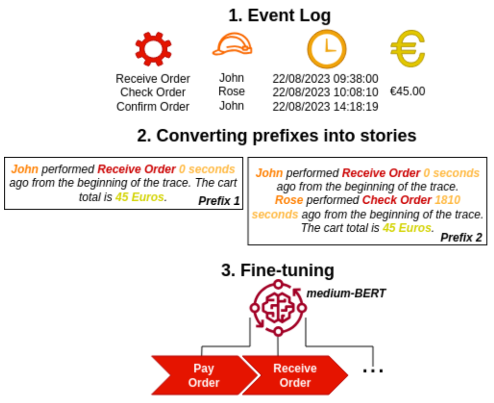

1. Event Log
Receive Order
Check Order
Confirm Order
John
Rose
John
22/08/2023 09:38:00
22/08/2023 10:08:10
22/08/2023 14:18:19
€45.00
2. Converting prefixes into stories
John performed Receive Order 0 seconds ago from the beginning of the trace. The cart total is 45 Euros.
Prefix 1
John performed Receive Order 0 seconds ago from the beginning of the trace. Rose performed Check Order 1810 seconds ago from the beginning of the trace. The cart total is 45 Euros.
Prefix 2
...
3. Fine-tuning
medium-BERT
Pay Order
Receive Order
...

Fig. 1. LUPIN pipeline

III. Proposed Approach

In this section, we illustrate LUPIN, a PPM approach that represents running process instances recorded in an event log as semantic stories of a business process and analyses these stories to address the task of the activity suffix prediction. Figure 1 shows the pipeline of LUPIN.

A. Basics

Let us consider an event log $\mathcal{L}$ that records traces of process instances executed for a business process $\mathcal{P}$. Each trace comprises optional global trace characteristics (e.g., the age of a patient involved in the hospitalization process) and a finite sequence of events, denoted as $\sigma = \langle e_1, e_2, \ldots, e_n, \perp \rangle$, where $\perp$ represents the end of a trace. An event contains mandatory local event characteristics such as activity and timestamp and optional characteristics such as resource or role. For each trace $\sigma \in \mathcal{L}$ with length $n+1$, we consider n distinct partitioning of σ in a prefix trace and suffix trace. Let us consider $1 \leq k \leq n$, $hd^k(\sigma) = \langle e_1, \ldots, e_k \rangle$ is the prefix of σ with size k, while $tl^{n-k+1}(\sigma) = \langle e_k + 1, \ldots, e_n \rangle$ is the suffix of σ with size $n - k + 1$, so that $\sigma = hd^k(\sigma) \cdot tl^{n-k+1}(\sigma)$ with $\cdot$ denoting the concatenation operator.

B. Transforming prefix traces into semantic stories

The transformation of each prefix trace into a semantic story is performed according to a narrative template that defines a semantic context for both global trace characteristics and local event characteristics. An example of prefix traces' generation of stories done according to a narrative template is shown in Figure 1. This example uses a narrative template formulated for an e-commerce event log that contains the global trace characteristic: cart value, and the local event characteristics: activity, resource and timestamp. The example narrative template is defined as follows: *"**[resource]** performed **[activity]** **[time_since_the_beginning_of_the_trace]** seconds ago from the beginning of the trace. The cart total is **[value]** Euros"*.

This narrative template includes non-terminal symbols to represent both trace and event characteristics in meaningful sentences so that the semantic stories can be easily obtained from prefix traces by replacing each non-terminal symbol of the narrative with the value of the corresponding characteristic in the prefix trace. Hence, the semantic story of a prefix trace includes a sentence for each event, which is generated according to the local narrative template of the event by replacing non-terminal event symbols with event characteristics' values. The event stories are produced from the latest event to the newest event recorded in the prefix trace. In addition, the semantic story of a prefix trace includes a final sentence to describe the global story of the trace characteristics. Again this sentence is generated according to the global narrative template of the trace by replacing non-terminal trace symbols with trace characteristics' values.

Final remarks concern the fact that several BERT models, comprising medium BERT, are subject to a maximum number of tokens permitted in the stories. In particular, medium BERT can input stories up to 512 tokens long. To handle this issue, we apply a truncation strategy that keeps the last 512 tokens considering the most recent events recorded in the prefix trace. This truncation strategy bases on the idea that the latest events recorded in a prefix trace are the most relevant to predict the future of a running trace [4].

C. LUPIN

Given the event log $\mathcal{L}$, every prefix trace recorded in $\mathcal{L}$ is transformed into a semantic story that is labeled with the sequence of activities observed in the corresponding suffix within the same trace. These labeled data are used to learn an LLM-based multi-output deep neural model to predict the activity suffix of a running trace so that the i-th activity of a suffix corresponds to the i-th target of the deep neural model. As the suffix of extracted from $\mathcal{L}$ may have different length, we apply the padding strategy that fills suffices with $\perp$ (end of trace) to achieve the fixed number of targets for the multi-output prediction. We set the maximum length of a suffix equal to the maximum length of a trace record in $\mathcal{L}$. Based on these premises, LUPIN adds a number of classification heads that is equal to the maximum suffix length to the pre-trained medium BERT architecture. As described in [17], medium BERT follows the Transformer architecture of BERT [23] defined according to a bi-directional approach by considering both the left and right context of words in a sentence. It includes 8 hidden layers (transformer blocks) and 8 self-attention heads with the hidden embedding size set equal to 512 and the filter size set equal to 2048 for a total of 41.7M parameters. Each transformer block is an encoder block with self-attention layers. In LUPIN, we use the pre-trained version of medium BERT that was made available by authors of [17] and obtained distilling a large teacher using its soft labels. The distillation teacher was also a Transformer, but composed of 24 hidden layers and 16 self-attention heads with the hidden embedding size set equal to 1024 and the filter size equal to 4096 for a total of 340M parameters. For each event log $\mathcal{L}$, the pre-trained medium BERT was fine tuned to the prefix stories recorded in $\mathcal{L}$ and labeled with the activity suffix, so that LUPIN can adapt the LLM to predict the activity suffix of a running instance for all output targets simultaneously. The fine tuning process allows us to update the parameters of the pre-trained medium BERT model based on gradients computed from multiple loss functions. The fine tuning is completed by optimizing the sum of the cross-entropy loss functions measured for the multiple targets on the training data.

D. Integrated Gradients

Integrated Gradients (IG) [18] is a local, XAI technique to explain the effect of input dimensions on deep neural model decisions by accounting for gradient knowledge. First, given a story, IG selects a baseline vector of zeros (one zero for each text token in the story). Then it computes the gradient of the model prediction for each input feature of the sample story to explain. This gradient represents how much the model prediction changes as the value of that token changes in the sample story. Gradients calculated for input dimensions are integrated along the straight-line path (network layers) from the baseline vector to the actual input. The integration is typically approximated by adopting techniques like Riemann sum or Gauss-Legendre quadrature rule. In this way, we explain the relevance of each input token based on the integrated gradient values computed. The higher the integrated gradient value measured for an input token, the more the prediction of the model is sensitive to changes in the value of the considered token. LUPIN uses IG to compute the gradient value for each input token in the semantic story of the running instance trace and for each distinct target of the suffix to predict.

E. Implementation details

LUPIN was implemented in Python 3.9.18–64 bit version using Torch 2.1.1 as the back-end. The source code of the proposed approach is available on the GitHub repository[2]. The pre-trained version of medium BERT used in the evaluation study is available on Hugging Face repository[3]. For the fine tuning process, we adopted the AdamW optimizer with learning rate equal to 1e-5 and batch size equal to 8. A maximum number of 50 epochs were run for each experiment. The training stopped before and if validation loss did not decrease for five consecutive epochs. The model with the best validation loss was saved and validated on the testing set. Finally, the code of LUPIN integrated the LayerIntegratedGradients version of IG that is available in Captum library[4].

IV. Experimental Results

In this section we illustrate the evaluation study conducted to assess the accuracy and explanation performance of LUPIN. Experiments were run on HP-Obelisk-Desktop with Intel(R) Core(TM) i7-9700F CPU @ 3.00GHz, NVIDIA GeForce RTX 2080, 32GB RAM and Ubuntu 20.04.6 LTS.

[2]https://github.com/vinspdb/LUPIN
[3]https://huggingface.co/prajjwal1/bert-medium
[4]https://captum.ai/api/layer.html

A. Event logs

We used six benchmark event logs available on the 4TU Centre for Research[5], except for the MIP dataset that was presented and published in [24]. In particular, we considered: three event logs (i.e., Helpdesk, BPI17O and BPI20R) following the Pareto's principle in the activity variants and three benchmark event logs (i.e., BPI15_1, MIP and Sepsis) not-following the Pareto's principle in the activity variants recorded in the log. We motivate this log selection considering that the Pareto principle states for many real-world phenomena where "a large portion of outcome come from a small number of causes". As illustrated in [25], the activity distribution of several real-life processes often follows the Pareto principle with a large portion of log activity traces held by a small fraction of top-frequent activity variants. On the other hand, [25] also shows that for larger processes with more activities and longer traces, the Pareto principle may no longer be observed with the most activity traces that are unique. Therefore, we selected both event logs following the Pareto principle and logs not-following the Pareto principle to explore the performance of our approach and related methods in both scenarios.

Table I reports the details of the event logs considered for the experiments. Specifically, BPI15_1 [26] contains all building permit applications recorded by five Dutch municipalities over a period of approximately four years. BPI17O [27] contains offers made for an accepted application through the online system of a Dutch financial institute in 2016 and their events until February 1, 2017. BPI20R [28] concerns requests for payment, which were not related to travels. Requests were collected for two departments in 2017, and the entire university in 2018. Helpdesk [29] concerns the ticketing management process of the Helpdesk of an Italian software company. MIP [24] records the simulation of a Management Incentive Program performed to resemble a real-world HR use-case in the conversational RPA domain. SEPSIS [30] records pathways of sepsis infections from a hospital.

B. Experimental setup

We adopted the experimental setting described in [7]. We sorted the traces of each event log by the timestamp of first events and selected the first 2/3 of the traces for the training stage, and the remaining 1/3 of the traces for the evaluation of the PPM model on the unseen traces. In the evaluation stage, we used the Damerau-Levenstein (DL) algorithm to measure the accuracy performance of the activity suffix predictions. This algorithm was adopted in several activity suffix prediction studies (e.g., [8], [11], [21]). It measures the distance between sequences by counting the number of edits needed for one string to match another. It penalizes actions like insertion, deletion, substitution, and transposition. These measurements are scaled by the maximum size of the compared sequences. In the experiments, we predicted an activity suffix for each running instance corresponding to a prefix of every testing trace and measured the DL distances between the predicted suffix and the corresponding actual suffix recorded in the trace. Subsequently we considered the inverse of this metric to gauge the similarity measurement. Finally, we computed the mean of similarity values measured for all running instances. The higher the average similarity value measured on the testing set, the better the accuracy performance of the PPM model.

[5]https://data.4tu.nl/portal

C. Related methods

We compared the performance of LUPIN to that of the multi-output version of the powerful LSTM-based method described in [3]. Similarly to LUPIN, the method described in [3] can process an input space that accounts for any global trace characteristic and any event trace characteristic recorded along a running process instance. Each data view defines a distinct input in a multi-input architecture. However, the method described in [3] is unable to take into account the semantic context that can be provided with a narrative skeleton. In addition, this PPM method was originally defined for the next-activity prediction task. So, keeping the running instance representation described in [3], we extended this method by replacing the single-output decision stage trained for the next-activity prediction with a multi-output decision stage trained for the activity suffix prediction. In the following, we denote this baseline method as LSTM. In addition, we compared LUPIN to three deep neural related methods [8], [11], [21] described in the recent PPM literature for the activity suffix prediction task (see details on these methods in Section II). For [8], we considered the two configurations described by the authors: ArgMax – AM (with the most likely activity selected at each step of the prediction) and Random Sampling – R (with the next-activity randomly sampled from the probability distribution of the next-activities). Finally, we considered an activity suffix prediction method that does not rely on deep learning. This was done to evaluate advantages of deep learning in the PPM task considered. To this aim, we considered activity suffix prediction models that is a multi-output Random Forest (RF) trained on a fixed-length feature vector representation of prefix traces extracted through the Aggregation encoding schema. We selected the Random Forest as machine learning algorithm and the Aggregation as encoding schema according to the results of an extensive comparative analysis including several machine learning algorithms and several encoding schema illustrated in [1]. The multi-output model was trained with one target for each activity in a suffix. To provide a fair comparison, we ran the related methods by considering the same experimental setup described in Section IV-B. We were able to run all the compared methods in the same experimental setting since the authors of the related methods provided the code. We set the parameters of the related methods according to the best parameter setup determined in the authors' provided code and described in the code repositories.

D. Results and discussion

We start this analysis by comparing the overall performance of the evaluated PPM methods. Table IV-D reports the average

TABLE I
EVENT LOG DESCRIPTION: NUMBER OF TRACES (♯**TRACE**), NUMBER OF EVENTS (♯**EVENT**), NUMBER OF ACTIVITY VARIANTS (♯**VARIANTS**), NUMBER OF THE TOP-REQUENT ACTIVITY VARIANTS CONTAINED IN THE 80% OF THE EVENT LOG (♯**PARETOVARIANTS**), NUMBER OF ACTIVITIES (♯**ACTIVITIES**), MINIMUM LENGTH OF TRACES (**MIN LENGTH**), MAXIMUM LENGTH OF TRACES (**MAX LENGTH**), MEAN LENGTH OF TRACES (**MEAN LENGTH**), NUMBER OF GLOBAL TRACE CHARACTERISTICS (**# TRACE CHARACT.**) AND NUMBER OF LOCAL EVENT CHARACTERISTICS (**# EVENT CHARACT**)

Event Log	#Trace	#Events	#Variants	#ParetoVariants	#Activities	Min Length	Max Length	Mean Length	#Trace Charact.	#Event Charact.
BPI15_1	1199	52217	1110	935	398	2	101	44	6	5
BPI17O	42995	193849	16	3	8	3	5	5	5	4
BPI20R	6886	36796	89	4	19	3	22	5	3	4
Helpdesk	4580	21348	226	7	14	2	15	5	2	8
MIP	1000	49604	1000	800	36	20	95	50	-	8
Sepsis	1050	15214	846	640	16	3	185	14	24	6

TABLE II
AVERAGE DL COMPUTED FOR THE BASELINE LSTM, THE RELATED DEEP LEARNING METHODS PRESENTED IN [8], [21] AND [11], THE MACHINE LEARNING METHOD RF AND LUPIN ON THE ACTIVITY SUFFIX PREDICTED FOR THE TESTING RUNNING INSTANCES RECORDED IN THE EVENT LOGS. EVENT LOGS THAT SATISFY THE PARETO PRINCIPLE ARE UNDERLINED. THE BEST RESULTS ARE IN BOLD, WHILE THE RUNNER-UP RESULTS ARE MARKED WITH *.

Method	BPI15_1	<u>BPI17O</u>	<u>BPI20R</u>	<u>Helpdesk</u>	MIP	Sepsis
LSTM	32.6	83.6	92.9*	81.1	40.6	43.9
[21]	35.8	73.9	92.2	85.2*	28.9	43.1
[8]R	33.2	74.8	79.1	74.0	42.1*	27.8
[8]AM	22.9	78.9	88.3	79.3	28.3	5.0
[11]	6.2	47.9	21.4	43.5	33.2	21.9
RF	36.6*	85.3*	88.1	84.1	38.4	45.9*
LUPIN	**42.1**	**86.0**	**93.6**	**85.9**	**49.2**	**47.0**

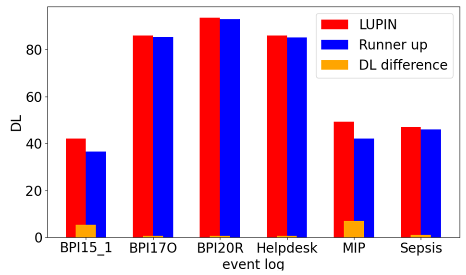


Fig. 2. Differences between the average DL computed for LUPIN and the runner-up method in Table IV-D

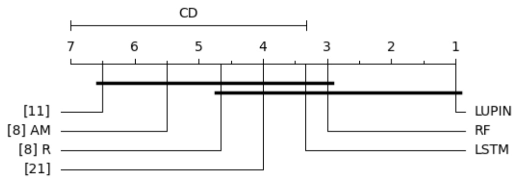


Fig. 3. Comparison among LUPIN and the related methods in terms of DL with the Friedman-Nemenyi test (computed $pvalue = 8.53e-05$). Groups of methods are not significantly different (at $p \leq 0.05$) are connected. CD denotes the critical difference area of the test.

DL metric measured for the baseline LSTM, the related methods described in [21], [8] – configuration R, [8] – configuration AM and [11], the machine leanring method RF and our method LUPIN. The DL metric is computed on the activity suffix predicted for all the testing running instances recorded in the study event logs. These results show that LUPIN outperforms, in average, all the remaining methods of this evaluation study by showing that representing running process instances as semantic stories allows us to gain accuracy in the activity suffix prediction problem. RF is the runner-up in three out of six event logs (i.e., 2 Pareto-compliant logs and 1 Pareto non-compliant log). In any case, the superiority of the performance of LUPIN on both deep learning and machine learning related methods is observed in all event logs. However, as shown in Figure 2, the gain in the average DL is commonly greater in the non-Pareto event logs than in the Pareto event logs. The critical difference diagram reported in Figure 3 also supports this conclusion. For this statistical comparison, we used the Friedman's test followed by the post-hoc Nemenyi test. This compares and ranks the average DL measured for the compared methods on multiple event logs. The test output shows that a statistical difference arises between the group of methods: LUPIN, RF and LSTM, and the group: [21], [8] – configuration R, [8] – configuration AM and [11]. It shows that LUPIN is the overall top-ranked method in the difference diagram with RF and LSTM as runner-up. Notably, the three top-ranked methods are the methods of this comparative study that adopt the multi-output learning strategy to predict the entire activity suffix in one step.

We proceed this accuracy performance analysis by examining the earliness of the activity suffix predictions yielded with both LUPIN and the related methods of this study. Figure 4 shows the average DL computed along the different prefix lengths for all testing traces recorded in the study event logs. These plots show that LUPIN outperforms related methods in almost all stages in five of six event logs and in the early stages of Sepsis event log. This is a desirable outcome, since it is highly demanded to yield accurate predictions of the evolution of a running instance in the early stages of a trace, to promptly discover possible anomalous or unexpected behaviors.

To complete the accuracy performance analysis, we analyse the Overall Accuracy (OA) measured on the single activities of all activity suffix predictions yielded for the testing running instances recorded the example event logs BPI20R and MIP.

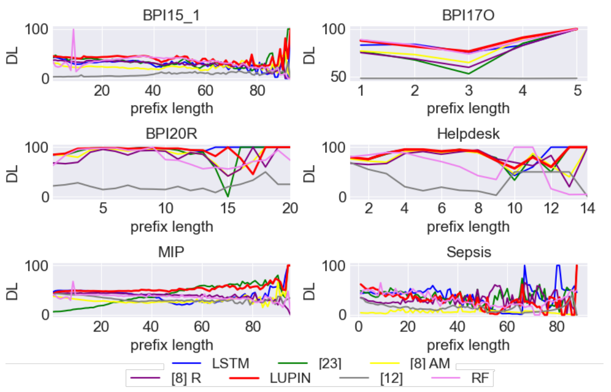


Fig. 4. Average DL (axis Y) measured on the testing running instances recorded in the study event logs and grouped across different running instance lengths (axis X)

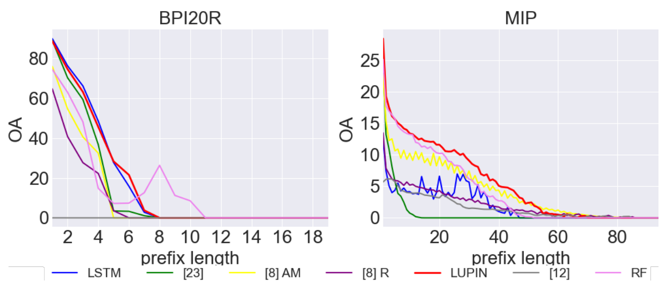


Fig. 5. OA (axis Y) on BPI20R and MIP event logs across different running instance lengths (axis X).

We selected these two event logs for this analysis considering that BPI20R is an event log compliant with the Pareto principle, while MIP is an event log non-compliant with the Pareto principle. Figure 5 shows the OA measured on the predictions yielded for the activities that respectively occupy the first position, the second position, and so on, in each predicted activity suffix. The plots produced for both BPI20R and MIP show that, as expected, the OA decreases as the predicted activity moves away from the last event actually observed in the running process instance. LUPIN still outperforms the left-out methods also in this analysis dimension. Again, the gain in accuracy is higher in the non-Pareto compliant log – MIP, than in the Pareto compliant log – BPI20R. In BPI20R, LSTM performs closely to LUPIN. Both methods use information recorded in all views of the training event log to train a multi-output deep neural model that predicts the entire activity suffix in one step. However, LUPIN leverages a LLM on the semantic history representation of running instances. This is particularly useful to gain accuracy in the non-Pareto event logs in all the steps of the predicted suffices.

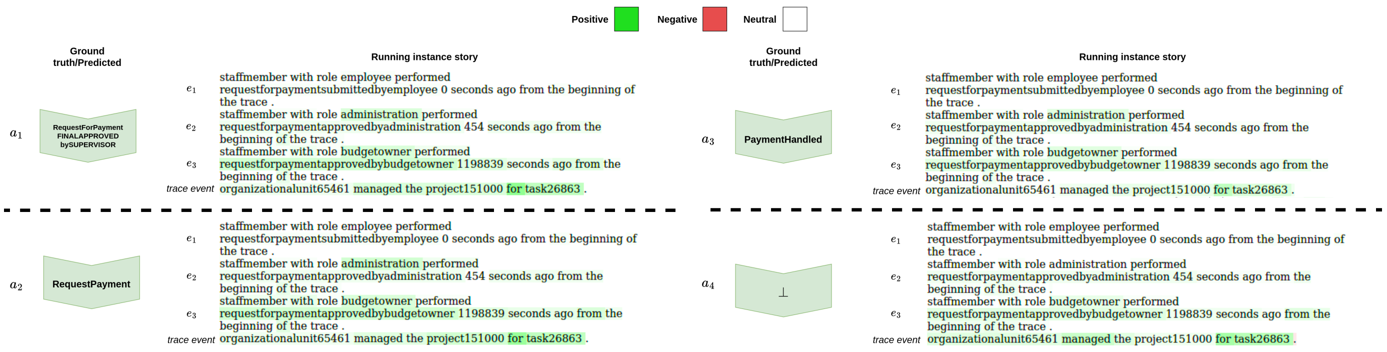
At the completion of this study, we illustrate an example of of the IG explanation produced from LUPIN for the activity suffix predicted for a running instance composed of three events and recorded in the testing event log of BPI20R. Figure 6 shows the activities of the suffix as they were predicted from LUPIN, the ground truth activities of the same suffix, and the explanation of the activity suffix prediction produced with IG. This explanation highlights in green which tokens of the semantic story of the running instance have a positive effect on each activity predicted in the suffix. Neutral tokens are colored in white, while tokens with a negative effect on the prediction are colored in red. The color intensity is proportional to the importance (in the positive or negative direction) of the effect of the text token on the specific prediction. As expected, the text stories of the most recent events observed in the running instance contribute to the prediction of each activity of the suffix more than the oldest events recorded in the running trace. This is noted regardless of the position of the predicted activity in the suffix. In addition, the role of ”administrator” recorded in the second event and the role of ”budget owner” recorded in the third event of the running trace are both important for the prediction of all the activities of the example suffix except for the $\perp$ activity. On the other hand, the importance of the activity ”request for payment approved by budget owner” has high importance for the prediction of the first two activities of the example suffix, while this activity loses importance for the prediction of the last two activities of the same example suffix. Final comments regard the global trace characteristics. For example, the task ”26863” is equally important for predicting all activities in this suffix.

V. Conclusion

This work introduces a new PPM method to address the activity suffix prediction task. We have adopted a semantic story-based representation of running process instances, and a LLM-based deep neural architecture for performing the activity suffix prediction. The deep neural architecture adds multiple classification heads to the pre-trained medium BERT model, to predict all activities of an activity suffix simultaneously. The multi-output architecture is fine tuned on the semantic stories of the historical running instances observed for a business process and used to predict the activity suffix of any new running instance of the same process. The evaluation study shows that the proposed method outperforms several related methods developed in the recent PPM literature in the considered event logs. However, the gain in accuracy is higher in the event logs that do not conform to the Pareto principle. As future developments, we intend to extend the experimentation to new event logs and compare the performance of different LLMs in the same task. We plan to explore the performance of the proposed LLM-based approach in the outcome prediction task. Finally, we would explore the potential of the proposed approach framed in the object-centric event data view.

Acknowledgment

The work of Vincenzo Pasquadibisceglie was supported by PNRR project FAIR - Future AI Research (PE00000013), Spoke 6 - Symbiotic AI (CUP H97G22000210007) under the NRRP MUR program funded by the NextGenerationEU and in partial fullfilment of the research objectives of PROMETEO project (CUP H93C23000900005). The work of Donato



a_1 RequestForPayment FINALAPPROVED bySUPERVISOR

e_1 staffmember with role employee performed requestforpaymentsubmittedbyemployee 0 seconds ago from the beginning of the trace .
e_2 staffmember with role administration performed requestforpaymentapprovedbyadministration 454 seconds ago from the beginning of the trace .
e_3 staffmember with role budgetowner performed requestforpaymentapprovedbybudgetowner 1198839 seconds ago from the beginning of the trace .
trace event organizationalunit65461 managed the project151000 for task26863 .

a_3 PaymentHandled

e_1 staffmember with role employee performed requestforpaymentsubmittedbyemployee 0 seconds ago from the beginning of the trace .
e_2 staffmember with role administration performed requestforpaymentapprovedbyadministration 454 seconds ago from the beginning of the trace .
e_3 staffmember with role budgetowner performed requestforpaymentapprovedbybudgetowner 1198839 seconds ago from the beginning of the trace .
trace event organizationalunit65461 managed the project151000 for task26863 .

a_2 RequestPayment

e_1 staffmember with role employee performed requestforpaymentsubmittedbyemployee 0 seconds ago from the beginning of the trace .
e_2 staffmember with role administration performed requestforpaymentapprovedbyadministration 454 seconds ago from the beginning of the trace .
e_3 staffmember with role budgetowner performed requestforpaymentapprovedbybudgetowner 1198839 seconds ago from the beginning of the trace .
trace event organizationalunit65461 managed the project151000 for task26863 .

a_4 $\perp$

e_1 staffmember with role employee performed requestforpaymentsubmittedbyemployee 0 seconds ago from the beginning of the trace .
e_2 staffmember with role administration performed requestforpaymentapprovedbyadministration 454 seconds ago from the beginning of the trace .
e_3 staffmember with role budgetowner performed requestforpaymentapprovedbybudgetowner 1198839 seconds ago from the beginning of the trace .
trace event organizationalunit65461 managed the project151000 for task26863 .

Fig. 6. Explanation produced with IG for each activity of the activity suffix predicted for the trace of a running process instance composed of three events and recorded in the testing event log of BPI20R

Malerba is in partial fulfillment of the project MAD - The Additive Metamorphosis of Design (ARS01_00717, CUP: B82C20000250004) funded by MUR. The authors wish to thank anonymous reviewers for the helpful suggestions they provided to improve the paper.

References

[1] I. Teinemaa, M. Dumas, M. L. Rosa, and F. M. Maggi, "Outcome-oriented predictive process monitoring: Review and benchmark," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 2, pp. 1–57, 2019.

[2] I. Verenich, M. Dumas, M. L. Rosa, F. M. Maggi, and I. Teinemaa, "Survey and cross-benchmark comparison of remaining time prediction methods in business process monitoring," *CoRR*, vol. abs/1805.02896, 2018.

[3] V. Pasquadibisceglie, A. Appice, G. Castellano, and D. Malerba, "A multi-view deep learning approach for predictive business process monitoring," *IEEE Trans. Serv. Comput.*, vol. 15, no. 4, pp. 2382–2395, 2022.

[4] ——, "JARVIS: Joining adversarial training with vision transformers in next-activity prediction," *IEEE Transactions on Services Computing*, pp. 1–14, 2023.

[5] V. Pasquadibisceglie, G. Castellano, A. Appice, and D. Malerba, "FOX: a neuro-fuzzy model for process outcome prediction and explanation," in *3rd International Conference on Process Mining, ICPM 2021*. IEEE, 2021, pp. 112–119.

[6] F. Folino, G. Folino, M. Guarascio, and L. Pontieri, "Data- & compute-efficient deviance mining via active learning and fast ensembles," *Journal of Intelligent Information Systems*, Jan 2024.

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

[8] M. Camargo, M. Dumas, and O. G. Rojas, "Learning accurate LSTM models of business processes," in *International Conference on Business Process Management, BPM 2019*, ser. LNCS, vol. 11675. Springer, 2019, pp. 286–302.

[9] N. Di Mauro, A. Appice, and T. M. A. Basile, "Activity prediction of business process instances with inception CNN models," in *In Advances in Artificial Intelligence, AI*IA 2019*, ser. LNCS, vol. 11946. Springer, 2019, pp. 348–361.

[10] V. Pasquadibisceglie, A. Appice, G. Castellano, and D. Malerba, "Predictive process mining meets computer vision," in *Business Process Management Forum, BPM 2020*, ser. LNBIP, vol. 392. Springer, 2020, pp. 176–192.

[11] F. Taymouri, M. L. Rosa, and S. M. Erfani, "A deep adversarial model for suffix and remaining time prediction of event sequences," in *Proceedings of the 2021 SIAM International Conference on Data Mining, SDM 2021*, pp. 522–530.

[12] P. Ceravolo, S. B. Junior, E. Damiani, and W. M. P. van der Aalst, "Tuning machine learning to address process mining requirements," *IEEE Access*, vol. 12, pp. 24 583–24 595, 2024.

[13] A. Oved, S. Shlomov, S. Zeltyn, N. Mashkif, and A. Yaeli, "SNAP: semantic stories for next activity prediction," *CoRR*, vol. abs/2401.15621, 2024.

[14] M. van Luijken, I. Ketykó, and F. Mannhardt, "An experiment on transfer learning for suffix prediction on event logs," in *Business Process Management Workshops*. Springer, 2024, pp. 31–43.

[15] H. Chen, X. Fang, and H. Fang, "Multi-task prediction method of business process based on bert and transfer learning," *Knowledge-Based Systems*, vol. 254, p. 109603, 2022.

[16] I. Ketykó, F. Mannhardt, M. Hassani, and B. F. van Dongen, "What averages do not tell: predicting real life processes with sequential deep learning," in *Proceedings of the 37th ACM/SIGAPP Symposium on Applied Computing, SAC 2022*, 2022, p. 1128–1131.

[17] I. Turc, M. Chang, K. Lee, and K. Toutanova, "Well-read students learn better: The impact of student initializationon knowledge distillation," *CoRR*, vol. abs/1908.08962, 2019.

[18] M. Sundararajan, A. Taly, and Q. Yan, "Axiomatic attribution for deep networks," in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017*, vol. 70. PMLR, 2017, pp. 3319–3328.

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

[20] M. Selim, R. Zhou, W. Feng, and O. Alam, "Reducing error propagation for long term energy forecasting using multivariate prediction," in *Proceedings of 35th International Conference on Computers and Their Applications, CATA 2020*, ser. EPiC Series in Computing, vol. 69, 2020, pp. 161–169.

[21] E. Rama-Maneiro, P. Monteagudo-Lago, J. C. Vidal, and M. Lama, "Encoder-decoder model for suffix prediction in predictive monitoring," *CoRR*, vol. abs/2211.16106, 2022.

[22] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 7, pp. 2409–2429, 2020.

[23] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding," in *Proceedings of NAACL-HLT 2019*. Association for Computational Linguistics, 2019, pp. 4171–4186.

[24] S. Zeltyn, S. Shlomov, A. Yaeli, and A. Oved, "Prescriptive process monitoring in intelligent process automation with chatbot orchestration," vol. 3310, 2022, p. 49 – 60.

[25] W. M. P. van der Aalst, "On the pareto principle in process mining, task mining, and robotic process automation," in *Proceedings of the 9th International Conference on Data Science, Technology and Applications, DATA 2020*. SciTePress, 2020, pp. 5–12.

[26] B. van Dongen, "Bpi challenge 2015 municipality 1," 2015.

[27] ——, "BPI Challenge 2017 – Offer Log, 4TU.Centre for Research Data, Dataset," 2017.

[28] ——, "BPI Challenge 2020 – Request for Payment, 4TU.Centre for Research Data, Dataset," 2020.

[29] M. Polato, "Dataset belonging to the help desk log of an italian company," 2017.

[30] F. Mannhardt, "Sepsis cases - event log," 2016.