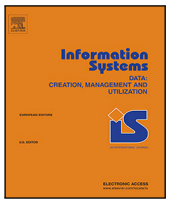
Information Systems 137 (2026) 102642

Contents lists available at ScienceDirect

Information Systems

journal homepage: www.elsevier.com/locate/is



Enhancing next activity prediction in process mining with Retrieval-Augmented Generation ☆

Angelo Casciani [a,*], Mario Luca Bernardi [b], Marta Cimitile [c], Andrea Marrella [a]

[a] Sapienza University of Rome, via Ariosto 25, 00185, Rome, Italy
[b] University of Sannio, Piazza Roma 21, 82100, Benevento, Italy
[c] Unitelma Sapienza, Piazza Sassari, 00185, Rome, Italy

ARTICLE INFO

Dataset link: https://github.com/angelo-casciani/rag_next_activity, https://huggingface.co/angeloc1/autotrain-phi-ft-nap

Keywords:
Predictive Process Monitoring
Next activity prediction
Large Language Model
RetrieVal-Augmented Generation

ABSTRACT

Next activity prediction is one of the main tasks of Predictive Process Monitoring (PPM), enabling organizations to forecast the execution of business processes and respond accordingly. Deep learning models are effective at predictions, but with the price of intensive training and feature engineering, rendering them less generalizable across domains. Large Language Models (LLMs) have been recently suggested as an alternative, but their capabilities in Process Mining tasks are still to be extensively investigated. This work introduces a framework leveraging LLMs and Retrieval-Augmented Generation to enhance their capabilities for predicting next activities. By leveraging sequential information and data attributes from past execution traces, our framework enables LLMs to make more accurate predictions without additional training. We evaluate the approach on a wide range of event logs and compare it with state-of-the-art techniques. Findings show that our framework achieves competitive performance while being more adaptable across domains. Moreover, we assess early prediction capabilities, validate the significance of observed differences through statistical testing, and explore the impact of fine-tuning. Despite these advantages, we also report the framework's limitations, mainly related to interleaving activity sensitivity and concept drifts. Our findings highlight the potential of retrieval-augmented LLMs in PPM while identifying the need for future research into handling evolving process behaviors and the development of standard benchmarks.

1. Introduction

Business processes are at the core of modern organizations, governing the execution of tasks and the flow of information across various domains. In this context, *Process Mining* [1] has emerged as a powerful discipline that leverages event logs to discover, monitor, and improve processes. Traditionally, process mining has been applied in a retrospective manner, focusing on process discovery, conformance checking, and model enhancement. However, the increasing availability of event data tracing the process executions and the advancements in machine learning and predictive analytics have led to the rise of *Predictive Process Monitoring* (PPM) [2], which extends conventional process mining techniques by enabling real-time predictions about the future of an ongoing case.

PPM aims to anticipate various aspects of a process instance, such as its final outcome, the remaining time until completion, or the sequence of future activities [3]. The ability to foresee these aspects can be highly beneficial for organizations, allowing them to take proactive measures to prevent undesirable outcomes, optimize resource allocation, and improve overall process efficiency [4]. Indeed, unlike traditional business process monitoring, which is typically reactive [5], PPM enables organizations to intervene before critical events occur.

Recent advancements in Artificial Intelligence and, specifically, in Large Language Models (LLMs) have opened new avenues for PPM research. LLMs have demonstrated remarkable capabilities in natural language understanding and generation, leading previous literature to explore their potential in Business Process Management and process mining [6–8]. In particular, LLMs can be leveraged for *next activity prediction*, a fundamental PPM task where the objective is to predict the next event in an ongoing process instance based on historical traces from an event log [7].

Our work explores the effectiveness of LLMs in next activity prediction through a *data-driven* approach, comparing different models in terms of predictive accuracy and robustness. Unlike traditional machine

☆ This article is part of a Special issue entitled: 'AI-Enhanced BPM' published in Information Systems.

* Corresponding author.
E-mail address: casciani@diag.uniroma1.it (A. Casciani).

https://doi.org/10.1016/j.is.2025.102642
Received 11 March 2025; Received in revised form 14 October 2025; Accepted 27 October 2025
Available online 3 November 2025

learning approaches that rely on structured event log data and feature engineering [9,10], as well as existing LLM-based methodologies already developed for this task [11], we propose an LLM-based framework that leverages *in-context learning* (ICL) and *retrieval-augmented generation* (RAG) to enhance the predictions by also considering the values of the attributes of the events in the traces. To the best of our knowledge, this is the first approach to next activity prediction that leverages RAG to enhance the predictions generated by the LLM. We have assessed the performance of various LLMs embedded in a PPM monitoring framework, considering multiple event logs from different domains.

This work investigates the following research questions (RQs) directed at evaluating the performance of the framework in different settings:

RQ1: *How effective is the approach on real and synthetic logs from various domains, and how does it compare with and without RAG?***RQ2**: *Is the approach effective compared to the baseline?***RQ3**: *How robust is the performance of the LLMs embedded in the proposed approach?*

By addressing these questions, we aim to provide a comprehensive evaluation of LLM-based next activity prediction and its potential implications for PPM. Specifically, we assess the performance of the framework on several (real-world and synthetic) event logs under multiple experimental configurations by varying the employed LLM, embedding model, and the bucketing strategy for the trace prefixes. We also provide a comparison with state-of-the-art techniques over a wide variety of event logs. Furthermore, we verified the statistical significance of the empirical findings through posthoc tests (namely, the Friedman and Nemenyi tests) and we investigated the earliness capabilities of the LLMs within the framework, offering an overview of how their predictive performance varies with different prefix lengths. Finally, we also explore the impact of fine-tuning on the next activity prediction task to discover how this can impact the overall performance and computational efficiency of the framework. The experimental results reveal that the proposed framework achieves high accuracy and macro-averaged performance, demonstrating also its robustness for the task, even avoiding costly fine-tuning processes for the employed LLM.

The rest of the paper is structured as follows. Section 2 provides the necessary background information, while Section 3 presents a detailed overview of related work to contextualize the proposed approach, which is introduced in Section 4. Section 5 describes the experimental setup and discusses the results. Section 6 examines potential threats to the validity of the study. Finally, Section 7 presents the conclusions.

2. Background

This section presents the necessary background information and definitions to facilitate understanding of the rest of the paper.

2.1. Event logs

Definition 1 (*Event*)**.** An event represents an atomic occurrence in a process execution. Formally, let $\mathcal{E}$ be the universe of all possible events. Each event $e \in \mathcal{E}$ is characterized by a tuple:

$$e = (case\_id, activity, timestamp, A_1, A_2, \dots, A_n)$$

where:

- $case\_id$ is a unique identifier of the process instance (case) to which the event belongs.
- $activity \in \mathcal{A}$ represents the activity associated with the event, where $\mathcal{A}$ is the set of all possible activities.
- $timestamp \in \mathbb{T}$ denotes the time at which the event occurred, where $\mathbb{T}$ is the set of all timestamps.
- $A_1, A_2, \dots, A_n$ are additional attributes providing further details about the event.

Definition 2 (*Trace*)**.** A trace is a finite sequence of events that represent the execution of a single process instance, i.e., a case. Formally, a trace σ is defined as:

$$\sigma = \langle e_1, e_2, \dots, e_m \rangle$$

where each event $e_i \in \mathcal{E}$ belongs to the same process instance and follows a strict total order based on the timestamps, i.e., $timestamp(e_1) < timestamp(e_2) < \cdots < timestamp(e_m)$.

Definition 3 (*Prefix Trace*)**.** Given a trace $\sigma = \langle e_1, e_2, \dots, e_m \rangle$, a prefix trace of length k ($1 \leq k < m$) is a truncated version of σ containing only the first k events:

$$prefix_k(\sigma) = \langle e_1, e_2, \dots, e_k \rangle.$$

The prefix trace represents the partial execution of a process instance.

Definition 4 (*Event Log*)**.** An event log $\mathcal{L}$ is a set of traces:

$$\mathcal{L} = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$$

where each trace σ_i corresponds to a distinct execution of the process, i.e., it contains all observed process instances.

2.2. Predictive process monitoring

PPM is a subfield of process mining focused on forecasting the future behavior of an ongoing (i.e., incomplete) process instance [2]. This includes making predictions about the outcome of a process execution, estimating the remaining time until completion, or forecasting the sequence of upcoming activities.

The ability to anticipate the outcome of a process instance, estimate its completion time, or determine the forthcoming activities is highly beneficial across various domains. For instance, early detection of potential deviations, failures, or delays in industrial and manufacturing processes enables organizations to take proactive measures, thus mitigating risks and optimizing process efficiency. Unlike traditional reactive monitoring techniques [5], where process violations or delays are only identified after they occur, predictive monitoring allows organizations to anticipate issues in advance and implement preventive strategies.

Driven by advancements in many research field such as Data Science, Predictive Analytics, and Artificial Intelligence, predictive methodologies have gained increasing popularity also within process mining and have become an essential component of modern process mining tools, supporting real-world applications in organizational decision-making [10,12].

PPM techniques generally rely on historical execution data to forecast the future course of an ongoing process instance. The predictive models used in this field typically operate in two phases: (*i*) a *training* phase, where models learn from historical, completed execution traces, and (*ii*) a *prediction* phase, where these trained models are queried to predict the future behavior of an active process instance.

Despite being a relatively young research area, PPM has experienced rapid growth in recent years [9] and research in this area can be categorized according to three main dimensions:

- *Prediction Type*: The nature of the predictions generated by the model.
- *Methodological Approach*: The computational techniques employed for making predictions.
- *Input Information*: The types of data used as input for the prediction's generation.

Regarding the type of predictions, prior research [9] has identified three primary categories:

- *Outcome-based predictions*: Forecasting predefined categorical or boolean outcomes of a process instance.

- *Numeric value predictions*: Estimating quantitative measures of interest, often represented as continuous or discrete numerical values.
- *Next activity predictions*: Predicting the sequence of forthcoming activities (possibly) along with their associated data attributes.

In this paper, we focus on proposing a new technique for the *next activity prediction* task.

2.2.1. Next activity prediction

Among the various predictive tasks in PPM, one of the most fundamental and widely studied problems is *next activity prediction* [9].

Next activity prediction relies on historical process execution traces to learn patterns in event sequences. Given a prefix trace extracted from an event log, the objective is to determine the most likely subsequent event based on past traces with similar behavioral patterns. This prediction is typically informed by the control-flow perspective, which considers solely the sequence of activities.

Formally, let $\mathcal{L}$ be an event log, and let $\sigma \in \mathcal{L}$ be a trace of that log, where $\sigma = \langle e_1, e_2, \dots, e_m \rangle$ is an ordered sequence of events. A prefix trace of σ is defined as $\sigma_p = \langle e_1, e_2, \dots, e_n \rangle$ with $n < m$, denoted as $\sigma_p \sqsubseteq \sigma$.

The goal of the next activity prediction task is to determine the next event e_{n+1} such that:

$$\sigma = \langle e_1, e_2, \dots, e_n, e_{n+1}, \dots, e_m \rangle.$$

The predicted event e_{n+1} must either be explicitly present in the event log $\mathcal{L}$ or derived as a result of a generalization starting from the log's traces.

2.2.2. Bucket strategies

In PPM, *bucket strategies* define how prefixes of traces are selected from full event logs to train and evaluate prediction models. Two common strategies are the *single bucket* approach, where all possible prefixes from each trace are used, and *gap-based* or *multi-bucket* approaches, which sample prefixes at regular intervals (e.g., every 2 or 3 events). While the single bucket strategy provides a fine-grained view of process execution over time, it can lead to large and unbalanced datasets, particularly when trace lengths vary significantly. To mitigate computational costs and reduce redundancy, gap-based strategies are often employed, as suggested in prior work [13]. These strategies aim to balance between data granularity and efficiency, making them well-suited for scalable predictive techniques.

2.3. Large language models

LLMs are computational models for Natural Language Processing (NLP) that provide powerful capabilities for understanding and generating human-like text [14]. Built on advanced *transformer-based* architectures, LLMs leverage self-attention mechanisms to capture relationships within input sequences and produce coherent output [15]. Models such as GPT [16], LLaMA [17], Mistral [18], and Phi [19], which comprise billions of parameters and are pre-trained on massive datasets, excel across several tasks from natural language question-answering [20] to code explanation and generation [21]. Recently, the Deepseek family of models represented a significant advancement in the development of efficient and scalable LLMs, particularly regarding their reasoning capabilities [22]. Moreover, by employing *knowledge distillation* techniques, these models manage to transfer the capabilities from larger models into smaller, more deployable architectures while maintaining competitive performance [22,23].

Despite their versatility, general-purpose LLMs are not optimized for domain-specific applications and can, thus, generate inaccurate information (i.e., *hallucinations*) [24]. Therefore, in specialized contexts and tasks, challenges arise due to the domain knowledge inherited from potentially outdated or biased training datasets [25].

Fine-tuning is commonly adopted to adapt LLMs to particular tasks. By training pre-trained models on task-specific datasets, fine-tuning enhances their performance and answers' relevance for specialized applications. However, the resource-intensive nature of this technique, including substantial computational and data requirements, can be a limiting factor [26]. In this scenario, a possible alternative is embodied by *in-context learning* (ICL) [27]. Leveraging carefully engineered prompts containing examples and explicit instructions, ICL allows LLMs to perform *zero-* or *few-shot learning*, guiding them to generate contextually appropriate responses without additional training.

ICL can be further strengthened by *Retrieval-Augmented Generation* (RAG), which improves the accuracy and reliability of LLM outputs, particularly in knowledge-intensive tasks [28]. RAG is an approach designed to enhance the capabilities of LLMs by combining their inherent knowledge with specialized, vast, and dynamic information coming from external knowledge repositories. RAG operates by retrieving relevant information based on similarity metrics with the user's input and integrating it into the prompt provided to the LLM, effectively leveraging ICL to utilize the additional knowledge.

In brief, RAG enriches the LLM's initial prompt by augmenting it with pertinent external information retrieved through search algorithms [25]. This additional context allows the model to produce more precise and contextually aware responses. RAG achieves this by incorporating a retriever and a generator into its architecture, following a three-step process: retrieve, augment, and generate.

- *Retrieve*: The user query x is used by the retriever $p_\eta(z|x)$, parameterized by η, to identify and retrieve relevant context (e.g., text documents z) from an external knowledge base. Using an embedding model, the query is embedded into a vector space, where it is matched against entries in a vector database. The retrieval step selects the k most similar documents based on vector similarity.
- *Augment*: The retrieved documents are combined with the original query to create an augmented prompt template. This enhanced prompt provides the LLM with the necessary contextual information to address the query more effectively.
- *Generate*: The retrieval-augmented prompt is input to the LLM, which uses the additional context to generate a response that reflects the enriched information from the retrieved chunks.

In this work, we rely on LLMs to perform the next activity prediction task by leveraging prompts enriched with RAG.

2.4. Embedding models

An *embedding model* is a machine learning model that converts discrete, categorical data (e.g., text, images, or, in this work, trace prefixes) into continuous vectors, typically in a high-dimensional space [29]. These models aim to capture the semantic relationships and underlying structure of the data.

Early approaches, such as traditional TF-IDF and static Word2Vec [30], focused on generating embeddings for individual words. However, modern, dynamic, and contextualized approaches leverage advanced architectures to create embeddings for entire sentences and paragraphs. Models based on the Transformer [15], such as BERT [31], use self-attention mechanisms to weigh the importance of different words in a sequence, capturing nuanced semantic relationships.

Sentence embedding models are optimized for tasks like semantic search, clustering, and paraphrase identification. For example, models from the Sentence-Transformers[1] library are widely used due to their efficiency and strong general-purpose performance. More recently, novel long-context embedding models, such as those by Jina AI [32],

[1] https://www.sbert.net/

have emerged for encoding longer sequences of tokens, making them particularly suitable for lengthy process execution traces.

Embedding models form the backbone of the RAG systems [25]. By converting both the user's query and the documents in a knowledge base into embeddings, the system can efficiently find the most relevant information using vector similarity search. In our work, this retrieved context, i.e., relevant past cases, is then used by an LLM to generate a more informed and accurate prediction about the next activity.

3. Related work

Next activity prediction is a critical task revealing business processes inefficiencies and driving process owner decisions [33]. For this reason, various approaches and numerous applications for predicting the next activity in a case have been introduced in the last decades. Previous studies focused on the adoption of traditional machine learning methods [34], considering numeric and categorical attributes. However, these methods struggled in terms of performance with sequential data, semantic, and free-text information [35].

To address these limitations, deep learning techniques have been widely explored in a business process [36–38], including Convolutional Neural Networks [39–41] and recurrent architectures such as Long Short-Term Memory (LSTM) networks. For instance, authors in [42] propose an LSTM-based approach to predict sequences of next events within their timestamp and the associated resources. However, the effectiveness of this approach is strongly reduced when there are activities with numerical attributes. Similarly, LSTM neural networks are used in [3] to perform several PPM tasks, demonstrating that LSTMs outperform alternative baseline techniques to predict the next activity and its timestamp on a running case. The method introduced by [36] is based on LSTM architecture to predict the next event in an ongoing trace. The authors note that it can be extended to predict other attributes, such as an event's duration. Specifically, categorical variables are represented using embedding techniques, transforming them into high-dimensional continuous vectors. The model consists of two hidden-layer LSTMs trained over one hundred epochs, with the input dimension varying based on the embedding representation.

An encoder–decoder framework to predict the next activity in an ongoing case is also introduced by Lin et al. [43], which leverages all available log information for prediction tasks and applies random embedding to represent each event and its attributes. Instead, a predictive process approach based on a combination of multi-view learning and deep learning is proposed in [44] to enhance accuracy by considering diverse information from event logs. The study describes several experiments using benchmark event logs to compare this approach's accuracy with existing methods in the literature. In contrast, [45] explored an adaptation of Generative Adversarial Networks (GANs) to sequential temporal data for the next event prediction. The training process consists of a two-player game where one neural network competes against another, resulting in predictions that are indistinguishable from ground truth. The authors highlight that the worst-case accuracy of their approach is at least equal to that of non-adversarial settings. Their experiments further demonstrate that this method improves prediction accuracy compared to baseline models and a simple network architecture. The analysis of the above-described studies shows that LSTM methods obtain good performance in the case of short sequential data but have reduced performance when free-text features, semantics, and long-term dependencies are considered.

More recently, the emergence of LLMs has inspired new studies based on their adoption in activity prediction tasks to overcome the above limitation. An evaluation of the ability of LLMs to solve semantics-aware PM tasks is proposed in [7]. The experiments show that LLMs struggle to tackle complex process mining tasks when there are few in-context examples. However, they perform well when fine-tuned to specific process tasks.

To enhance LLM-based predictions, researchers in [46] introduce LUPIN, an LLM approach to predict, on a running process, the next activity suffix by encoding past process instances in semantic text stories. The concept of process semantic stories is also explored in [11], which uses a new algorithm including semantic context information of the process event logs to predict the next activity. The method, called SNAP, compared with eleven state-of-the-art models, shows significant performance when applied to datasets with high numbers of semantic content. Finally, the authors in [47] introduce a PPM technique combining adversarial training with Vision Transformers to predict the next activity in a running process instance. This method considers multi-view information within a process trace, treating each view as a distinct patch of an image. Experiments on several benchmark event logs confirm the effectiveness of the approach compared to several state-of-the-art PPM techniques.

Differently from existing LLM-based methods which often focus on structuring process execution traces into semantic representations but overlook the data perspective, neglecting trace and event attributes in predictions, our framework integrates ICL and RAG, enabling predictions based on past process instances and data without requiring extensive fine-tuning.

Unlike traditional deep learning models, which rely on predefined feature engineering, our approach leverages LLMs' inherent ability to process and generalize patterns from data, enabling it to dynamically adapt to event logs of various natures and domains without requiring task-specific optimization.

Additionally, in contrast to the previous literature, our study takes a broader perspective by evaluating multiple LLM-based predictors across diverse event logs. This systematic comparison highlights their robustness and adaptability, offering new insights into the potential and limitations of LLMs in PPM.

4. Methodology

In this section, we present the overall methodology for predicting the next activity of a trace prefix using a RAG-based approach. Fig. 1 summarizes the complete framework. Our approach can be decomposed into two major parts:

- the *Event Log Preprocessing*, which happens offline, takes the XES event log in input and is responsible for its parsing to extract the formatted prefix traces and their storing as embeddings in a vector index; and
- the *Next Activity Prediction*, which happens online, receives as input the trace prefix over which performing the prediction of the next activity and it uses it to retrieve relevant past prefix traces from the vector store to leverage them for returning the prediction of the next activity.

4.1. Event log preprocessing

The *Event Log Preprocessing* phase takes place *offline* and is responsible for preprocessing the XES event log for the next activity prediction task. This step starts with the *Log Parser*, which extracts the traces from the XES event log in input while deleting eventual duplicates.

Once the log traces are obtained, the Log Parser generates possible prefixes based on a base number of events and a configurable gap between successive prefix lengths. The framework is designed to support multiple *bucket strategies*, from dense (i.e., including all possible prefixes) to more sparse sampling strategies based on fixed-length intervals. While including all prefixes can maximize the performance of the future predictions, it also introduces challenges such as exponential growth in the number of prefixes relative to the number of traces [48]. Each trace of length n generates up to $n - 1$ prefixes, leading to significant increases in storage and retrieval time within the vector index.

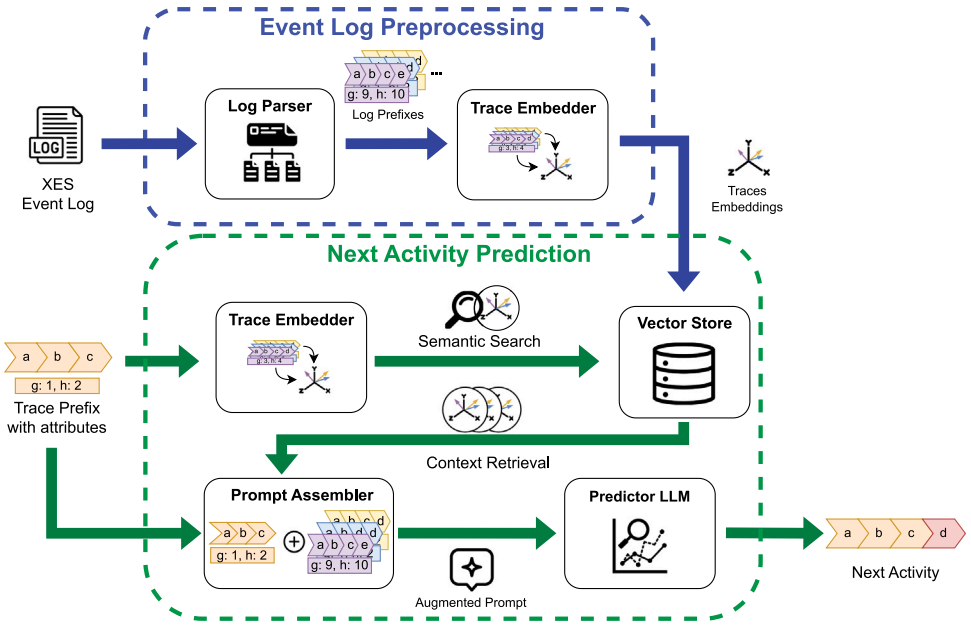


Fig. 1. The overall architecture of the framework.

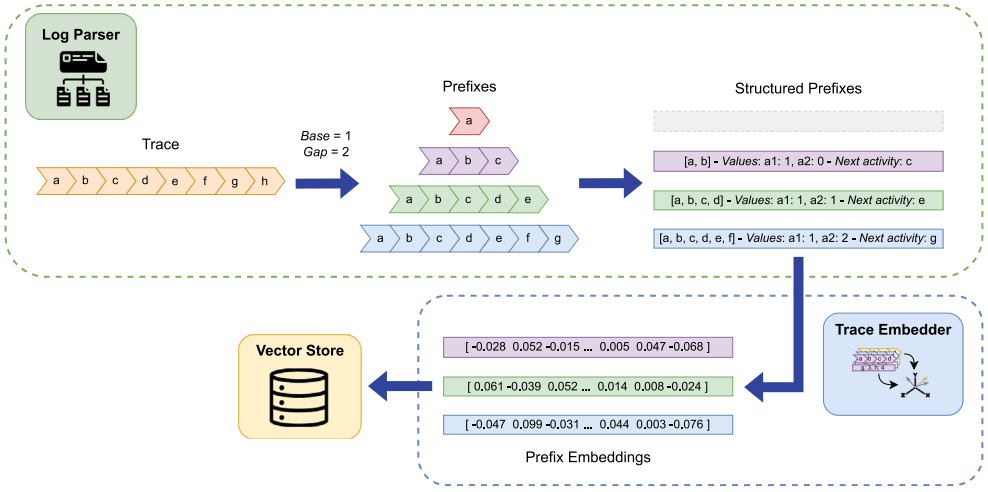


Fig. 2. The processing of a trace to produce the prefixes (in the example it is assumed a bucket strategy with base number of 1 and a gap of 2 events) is carried out by the Log Parser. The Trace Embedder computes the corresponding embeddings, which are then stored in the vector index.

Moreover, if the original traces differ greatly in length, longer cases will generate disproportionately more prefixes, potentially biasing the retrieval and LLM predictions toward these longer instances. To address these concerns, and in line with prior work such as [13], we also supported a gap-based filtering strategy, where only prefixes whose lengths equal a base plus multiples of a fixed gap (e.g., 1, 4, 7, 10 for a gap of 3) are retained, allowing for more granular control over the density of stored prefixes. In Section 5, we test the framework with different gap configurations, including the single bucket strategy (i.e., a gap of one event), to explore their impact on retrieval and prediction performance, still maintaining the XES formatting [49].

The remaining prefixes are further refined by the Log Parser (see Fig. 2) that removes unary prefixes (i.e., those consisting of a single activity) and produces, for each of them, a structured string containing: (*i*) the sequence of activity labels it contains up to the penultimate one, (*ii*) the final values of each attribute at the end of the prefix, and (*iii*) the name of the next activity following the prefix (i.e., the last activity included in it).

We chose to include only the final values of each data attribute for three main reasons. First, to remain within the LLM's context window. Second, to ensure semantic relevance during retrieval and avoid performance degradation associated with processing long numerical sequences. Indeed, embedding a large number of numerical values can hinder the effectiveness of similarity-based retrieval methods, as it may lead to the selection of chunks that are numerically similar but semantically irrelevant, thereby reducing the quality of predictions. Third, to prevent further amplification of this issue due to the "*lost in the middle*" phenomenon observed in long-context LLMs [50].

Such standardized prefixes are then passed to the *Trace Embedder*, which produces for each trace prefix the corresponding dense, low-dimensional vector representation (i.e., embedding) to be efficiently stored in a vector database and used in the following online stage to support the predictions.

4.2. Next activity prediction

Once the vector index has been constructed, containing the embeddings of all selected prefix traces from the event log, we proceed with

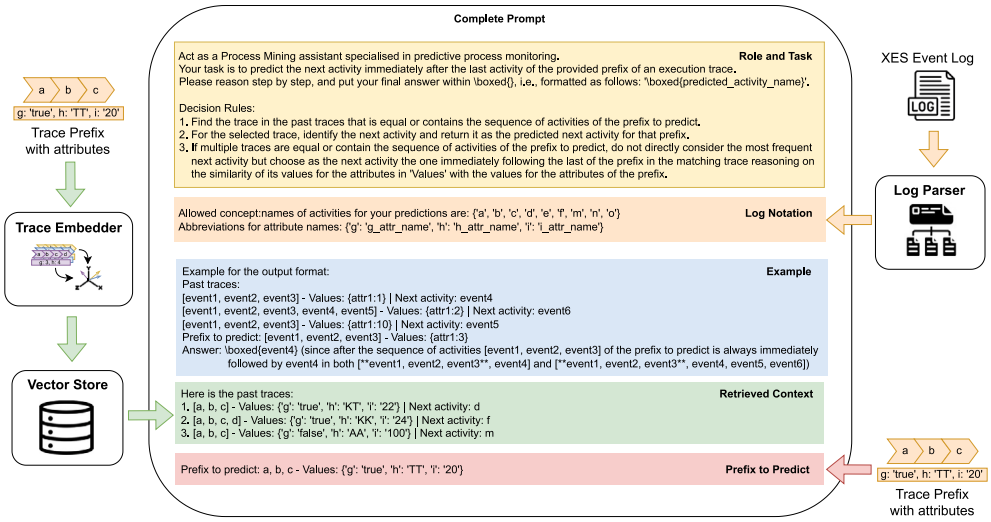


Fig. 3. The composition of the prompt for the LLM within the framework. The prompt is structured into five sections. The first three sections provide the LLM with its role, the specific prediction task, and the decision rules it should follow in making such a prediction. It also includes contextual information about the event log, such as the activity labels and attribute abbreviations, along with an illustrative example to ensure the model understands its task. The final two sections are dynamically generated and include the most similar trace prefixes retrieved from the vector store and the specific prefix for which a prediction is required.

the next activity prediction task. Given a prefix trace extracted from an execution of the business process recorded in the XES event log, our objective is to predict the next activity in the sequence.

The first step in this online stage is performed by the *Trace Embedder*, which takes the given prefix trace as input and computes its corresponding embedding. The embedding is then used to conduct a similarity search on the vector index based on *cosine similarity*, retrieving the top k most similar prefix traces from the past executions. During our experimentations, we found the value $k = 3$ to provide an optimal trade-off between retrieving a sufficient number of relevant prefixes to enhance prediction accuracy and mitigating the risk of the *lost-in-the-middle* phenomenon for longer contexts [50].

Once the most similar prefix embeddings are retrieved, they are mapped back to their corresponding event log traces. These retrieved traces serve as additional context for prompt engineering, enriching the input provided to the *Predictor LLM*. The Predictor LLM is responsible for generating the prediction of the next activity following the input prefix, leveraging both the retrieved historical information and the attributes associated with each trace.

The prompt[2] designed for the Predictor LLM follows the structure presented in Fig. 3. We employed several well-established prompt engineering techniques to ensure reliable answers [51]. These include explicitly defining the LLM role to clarify its task and output style [52], as well as incorporating an example of prefix-prediction pair to facilitate one-shot ICL [53].

The complete prompt includes five main sections: (*i*) the definition of the *LLM role* and the *specific task* it needs to perform within the framework, i.e., the prediction of the next event, along with the decision rules the Predictor LLM should follow in predicting the next activity based on the context; (*ii*) *contextual information about the event log* under analysis, i.e., the allowed labels for the activities and a legend for the abbreviations used for the attributes' names; (*iii*) a general *example* to guide the model in understanding how to perform the task; (*iv*) the *prefixes retrieved* from the vector index that have the highest similarity to the prefix to be predicted, based on both the sequence of activities and the values of the associated attributes; and (*v*) the *prefix over which performing the prediction*, handled by the LLM according to the previous parts of the prompt. It is important to note that (*i*) and (*iii*) are *fixed*, while (*ii*), (*iv*), and (*v*) are *dynamically generated*. Notably, (*ii*) is derived from log-specific information, (*iv*) is retrieved from the vector store through semantic similarity, and (*v*) is selected from the test prefixes on which the predictions are performed.

Specifically, Fig. 3 provides a complete example of how the prompt is constructed. The LLM is instructed to act as an assistant for PPM and is tasked with predicting the next activity for a given execution prefix. To do so, the LLM follows the decision rules provided in the system message: it should examine the retrieved past traces, identify the one most similar to the input prefix based on the sequence of activities and the final values of the attributes, and return the most likely next activity. These rules are then followed by the list of allowed activity names for the prediction and the abbreviations used for data attributes, introduced to reduce token usage within the LLM's context window. These elements are extracted by the Log Parser during the preprocessing stage. Next, the prompt includes a generic example using placeholder activity labels, helping the LLM understand the task and output format (e.g., `\boxed{event4}`). This is followed by the retrieved prefixes from the vector index, each accompanied by their corresponding attributes and next activity, serving as in-context examples. Finally, the prompt ends with the target prefix (e.g., $< a, b, c >$) and its last known values for the data attributes (g, h, and i). In the given example, the correct prediction is activity d, as the target prefix shares the same activity sequence with the first retrieved prefix, has the same boolean value for g (i.e., "*true*"), a similar string for h (i.e., "KT" and "TT"), and a close numerical value for i (i.e., 20 and 22).

4.3. Implementation details

The framework was implemented in Python 3.10 and designed to run in a 64-bit environment.

For the implementation of the *Event Log Preprocessing* step, we employed several embedding models for testing (further details are provided in Section 5). Regarding the vector store, we used qDrant,[3] an open-source vector database designed for storing and querying high-dimensional embeddings, in its containerized version via Docker[4]

[2] The prompt structure is available at https://github.com/angelo-casciani/rag_next_activity.

[3] https://qdrant.tech/

[4] https://www.docker.com/

for easy deployment. The connection to the vector index is managed through gRPC as the communication protocol.[5] Specifically, the choice of qDrant was also motivated by its ability to preserve privacy and maintain local data ownership, an essential requirement in many real-world industrial applications of PPM. In our setup, all prefix embeddings and related log data remain on the local machine, ensuring compliance with data governance and confidentiality constraints.

For the *Next Activity Prediction* module, we used again the qDrant vector store for indexing and retrieving embeddings of prefixes. We also employed LangChain[6] to support the interaction between prompt generation and the language model. As for the choice of the LLM, multiple models were tested, as described in the evaluation setup reported in Section 5.

It is important to note that the use of specific technologies (e.g., qDrant, Docker, LangChain) represents implementation-specific choices that serve to demonstrate the feasibility of our approach. These components can be substituted with alternative tools without affecting the overall architecture or methodology, which has been designed to be modular and extensible.

5. Experimental evaluation

This section discusses the experiments conducted to evaluate the correctness of the predictions returned by the methodology. To this end, in the following sub-sections, we describe the event logs used for the testing, the experimental setup, and the results.

5.1. Event logs

To quantitatively assess the proposed framework, we evaluated its performance on the next activity prediction task using six event logs comprising real-world and synthetic datasets.

5.1.1. Real-world event logs

For *real-world* event logs, we considered the *BPI Challenge 2012*, the *BPI Challenge 2013* (in the "*closed problems*" and "*incidents*" versions), the *BPI Challenge 2020* (specifically, its *International Declarations* version), the *Hospital Billing*, and the *Sepsis Cases* datasets.

The *BPI Challenge 2012* dataset captures the process of handling personal loan applications at a Dutch financial institution [54]. It consists of real-life event data extracted from an information system that supported the process, including milestones such as offer creation, acceptance, and finalization. The log includes a wide range of attributes associated with each event, such as organizational roles, timestamps, and resource identifiers, and it contains 262,200 events distributed across 13,087 traces. This dataset offers a suitable benchmark for evaluating PPM techniques thanks to the presence of multiple activities related to checks, approvals, and offers, especially in scenarios involving parallelism and complex decision logic.

The *BPI Challenge 2013* dataset originates from Volvo IT Belgium and captures the execution of incident and problem resolution workflows in IT service management processes by the VINST system. The dataset is partitioned into three sub-logs: the incidents log, the open problems log, and the closed problems log, each representing different perspectives of the management lifecycle. For the purposes of our work, we focus on the *closed problems* [55] and the *incidents* [56] logs. The closed problems dataset includes cases that have reached a formal resolution, providing well-defined process traces, while the incidents dataset reflects the more dynamic and responsive aspects of IT service handling.

The *BPI Challenge 2020 (International Declarations)* dataset contains event data related to international travel declarations, comprising 6,449 cases and 72,151 events [57]. It is a subset of the broader BPI Challenge 2020 dataset and spans two years of travel expense claims. The process begins when an employee submits a travel request, which is then forwarded to the travel administration for approval. If approved, the request is sent to the budget owner and subsequently to the supervisor. If the budget owner and supervisor are the same person, only one of these steps is required. In certain cases, the director must also approve the request. The process concludes either with the trip taking place or a payment request being submitted and processed. For international trips, additional supervisor approval is required, obtained by filing a travel permit, which must be approved before making any travel arrangements. To receive travel expense reimbursements, employees must file a claim. This can be done either immediately after incurring expenses (e.g., flight tickets or conference registration fees) or within two months after the trip (e.g., accommodation and meal costs, which are typically paid on-site).

The *Hospital Billing* event log was obtained from the financial modules of the ERP system of a regional hospital [58]. The event log contains events related to the billing of medical services provided by the hospital. Each trace of the event log records the activities executed to bill a package of medical services that were bundled together. The event log does not contain information about the actual medical services provided by the hospital. The 100,000 traces in the event log are a random sample of process instances that were recorded over three years. Events and attribute values have been anonymized. The timestamps of events have been randomized for this purpose, but the time between events within a trace has not been altered.

The *Sepsis Cases* event log captures real-life hospital cases of sepsis, a severe life-threatening condition typically caused by an infection [59]. Each case in the log represents a patient's treatment through the hospital. The events were recorded by the hospital's Enterprise Resource Planning (ERP) system, resulting in a dataset of approximately 1,000 cases and a total of 15,000 events covering 16 distinct activities. Additionally, 39 data attributes were logged, including details such as the responsible medical group, test results, and checklist information. To protect patient privacy, both event data and attribute values have been anonymized. While the event timestamps have been randomized, the relative time intervals between events within each trace remain unchanged.

5.1.2. Synthetic event logs

The *synthetic* event logs used in our evaluation include *SL2-2v-1r*, *SL3-multivariant*, and *Udon-ya*.

The *SL2-2v-1r* log includes 27 activities, comprising four gateways [33]. It defines two numeric attribute variables (A2.1 and A7.1), with only the first influencing gateway E3 by determining the subsequent activity based on non-overlapping intervals. The remaining two gateways are not affected by these variables.

The *SL3-multi-variant* log has been designed to generate a large number of process variants by leveraging combinations of multi-selection predicate sequences. This structure intentionally increases the behavioral complexity of the process model and challenges the ability of the model to predict the next activity in the context of a highly branched workflow. As shown in Fig. 4, a distinguishing feature of this log is the extensive presence of decision points, which significantly increase the number of possible different execution paths. The repeated introduction of bifurcations results in a complex process structure, where numerous alternative branches can be selected. This design leads to a high degree of trace dispersion, complicating the identification of dominant execution patterns. Given the finite attention window of current LLMs, this aspect may introduce difficulties in preserving a complete representation of all possible trajectories. The primary intent here is to assess the limitations of LLMs in predicting the next activity in scenarios with a rapidly expanding set of alternatives. The cumulative effect of previous choices influences the quality of the internal process

[5] https://grpc.io/docs/what-is-grpc/introduction/
[6] https://www.langchain.com/

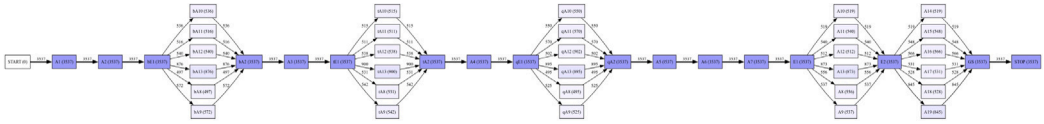


Fig. 4. Directly-follows graph of the SL3 multi-variant log.

representation, imposing constraints on the capacity of the model to maintain an accurate representation over an extended sequence.

These two synthetic logs were generated using a declarative log generator, introduced in [60], to produce event logs exhibiting the aforementioned properties needed for addressing the proposed research questions.

The *Udon-ya* dataset is a simulated event log in the XES format, representing operations in a fictional Japanese noodle restaurant [61]. The dataset contains 16,412 generated cases spanning one year, with a total of 674,486 events corresponding to 36 unique activities. It employs the lifecycle extension, providing start and complete events for each instance, along with scheduled events for a specific activity. Timestamps are realistically distributed according to daily and weekly business patterns. The underlying simulation model accounts for resource availability, capacity constraints, and queueing dynamics, which serve as the primary drivers of process behavior, alongside a moderately complex control flow. The process includes looping behavior, interleaved concurrency, and event attribute-dependent control-flow routing. Additionally, the dataset incorporates five instances of temporary concept drift with varying impact, introducing dynamic changes within the overall process timeline.

5.2. Experimental setting

We evaluated the framework in its ability to correctly predict the next activity in a trace prefix in the event logs presented previously.

To produce the testing dataset for each event log, we processed its traces with the Log Parser to produce the formatted prefixes as exposed before. We divided such a dataset into retrieval and testing subsets, allocating 70% and 30% of the prefixes respectively. The retrieval dataset was then stored in the vector index and used for context retrieval to aid the next activity prediction by the Predictor LLM. Conversely, the testing dataset was processed to generate, for each test sample, a tuple $< prefix, expected\_answer >$, where *prefix* represents the sequence of activities along with its attribute values and *expected_answer* corresponds to the correct next activity based on the prefix and its attributes. We selected 300 testing samples for each evaluation run.

To address **RQ1**, we assess the proposed methodology under various conditions. Specifically, we examine how RAG impacts the effectiveness of the predictions and whether the framework's performance is influenced by the nature of the event log, analyzing XES event logs from different domains and sources, including both real-world and synthetically generated logs. To assess the impact of prefix selection on storing, retrieval, and prediction performance, we experimented with different bucketing strategies for generating and storing prefixes in the vector index. In particular, we evaluated the performance of our framework using a gap of one activity (i.e., the single bucket strategy, which includes all possible prefix traces), as well as with gaps of three and five activities. This variation allows us to explore how denser or sparser prefix sampling influences the effectiveness of the prediction pipeline. Moreover, to better understand the role of embedding quality in retrieval-augmented next activity prediction, we conducted a study comparing several sentence embedding models. We included two widely adopted lightweight models from the Sentence-Transformers library: `all-MiniLM-L6-v2`[7] and `all-MiniLM-L12-v2`[8] (both with a maximum sequence length of 256 tokens and 384-dimensional vectors). These models offer fast inference and low memory usage, making them ideal for scenarios with limited computational resources or latency constraints. Comparing the 6-layer and 12-layer variants allowed us to quantify the performance gain achievable with increased model capacity at marginal additional cost. We also evaluated `all-mpnet-base-v2`[9] (384-token input, 768-dimensional vectors), an encoder known for its strong performance across semantic similarity benchmarks, offering a higher-capacity alternative within the same framework. To address the limitations of shorter context windows, we additionally tested three recent long-context models from Jina AI: `jina-embeddings-v2-base-en`[10] and `jina-embeddings-v3`,[11] both supporting inputs up to 8192 tokens, and `jina-embeddings-v4`,[12] which extends the input length to 32,768 tokens and produces 2048-dimensional embeddings. These models are specifically optimized for encoding lengthy textual inputs, allowing us to investigate whether retaining longer process histories leads to more accurate semantic representations, improved retrieval performance, and ultimately better predictive results.

On the other hand, **RQ2** seeks to evaluate the effectiveness of our approach in comparison with existing techniques from the literature for next activity prediction on the selected real-world event logs. However, it is important to highlight that PPM research currently suffers from a well-known lack of standardization in terms of evaluation methods, metrics, and even datasets. As noted in recent studies [48,62], different works often adopt different preprocessing procedures, which lead to substantial variations in the input data and experimental settings. Moreover, while event logs from the BPI Challenge are commonly used, they were originally curated for general process mining tasks, not always aligning with the goals of PPM. In this context, the metrics reported across different baseline methods tend to vary considerably, making it difficult to directly compare all approaches on a shared set of performance indicators. Consequently, we focused our comparative evaluation on the accuracy, which is shared across the selected baselines, and on approaches considering a single bucket strategy, ensuring a fair and meaningful comparison.

Finally, to tackle **RQ3**, we evaluate the robustness of the different LLMs embedded into the framework by examining the consistency of their performance in the next activity prediction task across various event logs. To reinforce this analysis, we performed post-hoc statistical tests, i.e., Friedman test [63] followed by Nemenyi test [64], to assess whether the observed predictive performance among the models are statistically significant, based on their mean scores for each metrics across the considered datasets, and to rank the best models within the framework. We assumed as *null hypothesis* H_0 to disprove that the LLMs within the framework deliver equal performance, while the alternative H_1 was that their performance differences are statistically significant. Additionally, we investigated the prediction accuracy of the three top-performing LLMs (according to the previous test) over different prefix length intervals to assess their *early prediction* capabilities. We considered the *BPI Challenge 2020 International Declarations* and *Sepsis Cases* logs as they are among the most representative in terms of prefix length variability, providing a wide coverage across all prefix length buckets considered in this evaluation. This was done

[7] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2
[8] https://huggingface.co/sentence-transformers/all-MiniLM-L12-v2
[9] https://huggingface.co/sentence-transformers/all-mpnet-base-v2
[10] https://huggingface.co/jinaai/jina-embeddings-v2-base-en
[11] https://huggingface.co/jinaai/jina-embeddings-v3
[12] https://huggingface.co/jinaai/jina-embeddings-v4

by grouping prefixes into progressive intervals, ranging from very early to very late in the trace, and computing the accuracy within each group. Such an assessment shows how the framework behaves in early phases when less information is available and how predictive accuracy evolves as the process execution unfolds. Furthermore, we also investigated the impact of fine-tuning on the overall performance of the framework. Specifically, we applied parameter-efficient fine-tuning using LoRA [65] to the best-performing open-source model in our evaluation, i.e., `phi-4`. Following the setup proposed in [7], we fine-tuned the model on the publicly available *S_NAP* dataset [66] using the AdamW algorithm [67], an initial learning rate of 1×10^{-5}, gradient accumulation over 16 batches to simulate an effective batch size of 32, and we run the training for three epochs. The average training time per epoch was 21.7 *hours*, in line with the ones in [66]. We made the model available online for testing.[13]

To assess the performance of the framework, we used four main metrics: *accuracy*, *precision* (macro), *recall* (macro), and *F1-score* (macro). *Accuracy* measures the proportion of correctly predicted instances over the total number of test prefixes, providing a straightforward assessment of overall correctness. By definition, accuracy ranges between 0 and 1. *Precision* (macro) computes the arithmetic mean of the precision values across all classes, ensuring that each class is given equal weight regardless of its frequency in the dataset. Similarly, *recall* (macro) is obtained by averaging the recall scores of all classes, treating them uniformly to prevent bias toward dominant classes. *F1-score* (macro) is the harmonic mean of precision and recall for a more balanced summarization of model performance. Specifically, this macro metric is calculated using the arithmetic mean of all *F1-scores* per class, treating all classes equally.

In our experimentation, we found that retrieving the *top three* most similar prefixes generates sufficient context for generating reliable predictions and avoids adding too many event log variants that might mislead the LLM.

To evaluate the generalizability of our approach across different language models, we tested a broad range of LLMs that differ in size, architecture, and training. Our selection includes primarily *open-source* models to improve transparency and replicability of our study, which are also crucial for the real-world adoption of PPM techniques in industrial settings. Specifically, we evaluated: `Llama-3.1-8B-Instruct`,[14] `Llama-3.2-1B-Instruct`,[15] and `Llama-3.2-3B-Instruct`[16] from MetaAI, `Mistral-7B-Instruct-v0.2`[17] and `Mistral-7B-Instruct-v0.3`[18] from MistralAI, `Qwen2.5-7B-Instruct`[19] from Qwen, `phi-4`[20] from Microsoft, `DeepSeek-R1-Distill-Qwen-7B`,[21] and `DeepSeek-R1-Distill-Llama-8B`[22] from DeepSeek. These models were chosen to represent architectural diversity (e.g., transformer depth and attention mechanisms), different instruction tuning strategies, and varying model capacities, from lightweight 1B models to more capable 7B/8B ones, allowing us to assess how model size impacts performance in the prediction task. We also included OpenAI's `gpt-4o-mini`[23] as a proprietary baseline, enabling us to compare open- and closed-source models.

The experiments were conducted on a workstation equipped with Linux/Ubuntu 22.04.3 LTS operating system and powered by an NVIDIA A100 GPU.

[13] https://huggingface.co/angeloc1/autotrain-phi-ft-nap
[14] https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct
[15] https://huggingface.co/meta-llama/Llama-3.2-1B-Instruct
[16] https://huggingface.co/meta-llama/Llama-3.2-3B-Instruct
[17] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2
[18] https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.3
[19] https://huggingface.co/Qwen/Qwen2.5-7B-Instruct
[20] https://huggingface.co/microsoft/phi-4
[21] https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Qwen-7B
[22] https://huggingface.co/deepseek-ai/DeepSeek-R1-Distill-Llama-8B
[23] https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/

5.3. Evaluation results

We now discuss the results of the quantitative evaluation of the framework in the next activity prediction task, according to the previously defined evaluation settings.

***Rq1**: how effective is the approach on real and synthetic logs from various domains, and how does it compare with and without rag?*

To address **RQ1**, we conducted an evaluation of the proposed methodology using the event logs and the settings introduced earlier. Additionally, we leveraged multiple language models to assess their performance within the approach, enabling a more comprehensive analysis.

With vs. Without RAG evaluation. First, we evaluated the impact of RAG on the next activity prediction task across different LLMs and event logs (i.e., the real-world *BPI Challenge 2020 (Int. Decl.)* and the synthetic *SL3-multivariant* event logs), with the results summarized in Table 1. As expected, incorporating contextual information significantly enhances performance. Specifically, shifting from a LLM that relies solely on task instructions and a single task example to one that also receives relevant context within the prompt, i.e., similar past trace prefixes from the event log, leads to a marked improvement in predictive accuracy for all the considered models and event logs.

Bucketing strategy evaluation. Thus, we proceeded with our assessment by varying the *bucketing strategy*, using the same event logs and the top-performing LLMs. In addition to the default setting with a gap of 1 (i.e., single bucket), we also tested gaps of 3 and 5 events. The results in Table 2 show that for the BPI Challenge 2020 (International Declarations) log, increasing the gap from 1 to 5 led to improved performance for `Qwen2.5-7B-Instruct`, while the other LLMs exhibited mixed results. The highest accuracy was achieved by `phi-4` with a gap of 3 (i.e., 0.94), although this is very close to the one obtained with the single bucket strategy (i.e., 0.93). Conversely, the results for the SL3-multivariant synthetic log show more modest performance, with accuracy ranging between 0.40 and 0.46 across all models and gaps, likely due to the high variability in the log. These findings highlight that the optimal gap may vary depending on the nature of the event log and the LLM's ability to generalize across different prefix variants. For this reason, in the next experiments, we adopted the single bucket strategy to ensure consistency with the majority of the approaches from the literature (like the baselines considered in **RQ2**).

Embedding model evaluation. Afterwards, we varied the embedding model to understand its impact on the prediction accuracy. Table 3 shows how the embedding model's choice significantly affects the performance, both in real-world and synthetic logs. For the *BPI Challenge 2020* log, compact transformer-based models such as `all-MiniLM-L6-v2`, `all-MiniLM-L12-v2`, and `all-mpnet-base-v2` consistently lead to strong performance, particularly with `phi-4` and using `all-MiniLM-L12-v2` (i.e., 0.93). Conversely, long-context embedding models like `jina-embeddings-v2-base-en`, `jina-embeddings-v3`, and `jina-embeddings-v4` underperform with both the LLMs, possibly due to lower fine-tuning alignment with the short sequences of trace prefixes. These findings are confirmed for the *SL3-multivariant* log: traditional sentence transformers achieve moderately good performance, while long-context models again report significantly lower accuracy. This suggests that long-context embedding capacity alone does not guarantee better retrieval quality, particularly when semantic similarity in short prefixes is pivotal. Based on these outcomes, we selected `all-MiniLM-L12-v2` as the embedding model for the subsequent evaluations, as it consistently achieved strong results, maintaining a good balance between performance and computational cost.

Table 1
Accuracy in the next activity prediction task for the *BPI Challenge 2020 International Declarations* (real-world) and *SL3-multivariant* (synthetic) log, as reported by the Language models operating within (with RAG) and independently (without RAG) of the framework. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively.

Language model	Event Log	RAG	Accuracy
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*		0.19
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	✓	*0.75*
phi-4	*BPI Challenge 2020 (Int. Decl.)*		0.02
phi-4	*BPI Challenge 2020 (Int. Decl.)*	✓	**0.93**
DeepSeek-R1-Distill-Llama-8B	*BPI Challenge 2020 (Int. Decl.)*		0.09
DeepSeek-R1-Distill-Llama-8B	*BPI Challenge 2020 (Int. Decl.)*	✓	0.70
Qwen2.5-7B-Instruct	*SL3-multivariant*		0.36
Qwen2.5-7B-Instruct	*SL3-multivariant*	✓	**0.46**
phi-4	*SL3-multivariant*		0.06
phi-4	*SL3-multivariant*	✓	*0.42*
DeepSeek-R1-Distill-Llama-8B	*SL3-multivariant*		0.09
DeepSeek-R1-Distill-Llama-8B	*SL3-multivariant*	✓	0.40

Table 2
Accuracy in the next activity prediction task for the *BPI Challenge 2020 International Declarations* (real-world) and *SL3-multivariant* (synthetic) log, as reported by the Language models operating with various *bucketing strategies* (i.e., gaps of events in a trace prefix). The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively.

Language model	Event log	Gap	Accuracy
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	1	0.75
phi-4	*BPI Challenge 2020 (Int. Decl.)*	1	*0.93*
DeepSeek-R1-Distill-Llama-8B	*BPI Challenge 2020 (Int. Decl.)*	1	0.70
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	3	0.81
phi-4	*BPI Challenge 2020 (Int. Decl.)*	3	**0.94**
DeepSeek-R1-Distill-Llama-8B	*BPI Challenge 2020 (Int. Decl.)*	3	0.64
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	5	0.85
phi-4	*BPI Challenge 2020 (Int. Decl.)*	5	0.91
DeepSeek-R1-Distill-Llama-8B	*BPI Challenge 2020 (Int. Decl.)*	5	0.70
Qwen2.5-7B-Instruct	*SL3-multivariant*	1	**0.46**
phi-4	*SL3-multivariant*	1	0.42
DeepSeek-R1-Distill-Llama-8B	*SL3-multivariant*	1	0.40
Qwen2.5-7B-Instruct	*SL3-multivariant*	3	**0.46**
phi-4	*SL3-multivariant*	3	*0.44*
DeepSeek-R1-Distill-Llama-8B	*SL3-multivariant*	3	0.40
Qwen2.5-7B-Instruct	*SL3-multivariant*	5	**0.46**
phi-4	*SL3-multivariant*	5	*0.44*
DeepSeek-R1-Distill-Llama-8B	*SL3-multivariant*	5	0.41

Table 3
Accuracy in the next activity prediction task for the *BPI Challenge 2020 International Declarations* (real-world) and *SL3-multivariant* (synthetic) log, as reported by the Language models operating with different *embedding models* for storing/retrieval of prefixes to/from the vector index. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively.

Language model	Event Log	Embedding model	Accuracy
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	*all-MiniLM-L6-v2*	0.80
phi-4	*BPI Challenge 2020 (Int. Decl.)*	*all-MiniLM-L6-v2*	*0.92*
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	*all-MiniLM-L12-v2*	0.75
phi-4	*BPI Challenge 2020 (Int. Decl.)*	*all-MiniLM-L12-v2*	**0.93**
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	*all-mpnet-base-v2*	0.76
phi-4	*BPI Challenge 2020 (Int. Decl.)*	*all-mpnet-base-v2*	0.90
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	*jina-embeddings-v2-base-en*	0.75
phi-4	*BPI Challenge 2020 (Int. Decl.)*	*jina-embeddings-v2-base-en*	0.88
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	*jina-embeddings-v3*	0.54
phi-4	*BPI Challenge 2020 (Int. Decl.)*	*jina-embeddings-v3*	0.55
Qwen2.5-7B-Instruct	*BPI Challenge 2020 (Int. Decl.)*	*jina-embeddings-v4*	0.53
phi-4	*BPI Challenge 2020 (Int. Decl.)*	*jina-embeddings-v4*	0.56
Qwen2.5-7B-Instruct	*SL3-multivariant*	*all-MiniLM-L6-v2*	*0.45*
phi-4	*SL3-multivariant*	*all-MiniLM-L6-v2*	0.44
Qwen2.5-7B-Instruct	*SL3-multivariant*	*all-MiniLM-L12-v2*	**0.46**
phi-4	*SL3-multivariant*	*all-MiniLM-L12-v2*	0.42
Qwen2.5-7B-Instruct	*SL3-multivariant*	*all-mpnet-base-v2*	**0.46**
phi-4	*SL3-multivariant*	*all-mpnet-base-v2*	**0.46**
Qwen2.5-7B-Instruct	*SL3-multivariant*	*jina-embeddings-v2-base-en*	**0.46**
phi-4	*SL3-multivariant*	*jina-embeddings-v2-base-en*	*0.45*
Qwen2.5-7B-Instruct	*SL3-multivariant*	*jina-embeddings-v3*	0.21
phi-4	*SL3-multivariant*	*jina-embeddings-v3*	0.27
Qwen2.5-7B-Instruct	*SL3-multivariant*	*jina-embeddings-v4*	0.17
phi-4	*SL3-multivariant*	*jina-embeddings-v4*	0.29

Table 4
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *BPI Challenge 2012* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.53	0.43	0.38	0.39	4.15	49.8
Llama-3.2-1B-Instruct	0.17	0.33	0.19	0.21	0.85	10.3
Llama-3.2-3B-Instruct	0.27	0.24	0.20	0.20	3.44	41.3
Mistral-7B-Instruct-v0.2	0.42	0.42	0.37	0.38	2.76	33.2
Mistral-7B-Instruct-v0.3	0.46	0.51	0.48	0.47	1.92	19.4
Qwen2.5-7B-Instruct	0.57	0.52	0.51	0.51	3.43	41.1
phi-4	*0.67*	*0.63*	*0.67*	*0.63*	3.39	40.7
DeepSeek-R1-Distill-Qwen-7B	0.47	0.45	0.44	0.44	4.21	50.5
DeepSeek-R1-Distill-Llama-8B	0.55	0.55	0.53	0.54	7.78	93.3
gpt-4o-mini	**0.76**	**0.72**	**0.68**	**0.67**	1.68	20.1

Table 5
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *BPI Challenge 2013 (Closed Problems)* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.72	0.70	0.71	0.70	3.54	42.4
Llama-3.2-1B-Instruct	0.17	0.17	0.16	0.16	1.21	14.5
Llama-3.2-3B-Instruct	0.28	0.26	0.27	0.26	1.96	23.5
Mistral-7B-Instruct-v0.2	0.42	0.37	0.37	0.36	2.98	35.7
Mistral-7B-Instruct-v0.3	0.66	0.55	0.57	0.55	0.52	6.25
Qwen2.5-7B-Instruct	0.83	0.78	*0.79*	0.78	3.52	42.1
phi-4	*0.89*	*0.85*	**0.87**	*0.86*	5.96	71.5
DeepSeek-R1-Distill-Qwen-7B	0.62	0.58	0.57	0.57	7.43	89.16
DeepSeek-R1-Distill-Llama-8B	0.67	0.63	0.62	0.63	14.68	176.16
gpt-4o-mini	**0.90**	**0.87**	**0.87**	**0.87**	1.13	13.6

LLMs and event logs evaluation. Eventually, we proceeded in our **RQ1** investigation with the evaluation of the framework's performance by varying the Predictor LLM. Table 4 presents the performance of the framework using different LLMs for the next activity prediction task on the real-world *BPI Challenge 2012* log. The `gpt-4o-mini` model registers the best results across all metrics, demonstrating the superiority of proprietary models for this event log within the framework. Among the open-source alternatives, the best model is `phi-4`, which outperforms all its peers while maintaining an acceptable inference time, comparable to that of the other LLMs with similar parameter size. On the other hand, smaller models such as the `Llama-3.2` variants are the fastest in terms of inference time but also report the weakest performance in the next activity prediction task. Finally, the distilled `Deepseek` models do not justify their highest computational costs with performance that aligns with the average of the other open-source models.

Table 5 shows the performance outcomes for the *BPI Challenge 2013 (Closed Problems)*. Overall, the closed-source `gpt-4o-mini` model achieved the best results across all metrics, while also maintaining one of the fastest average inference times. Nevertheless, among the open-source models, `phi-4` performed in a similar way, ranking second-best in all metrics and demonstrating solid generalization capabilities, albeit with higher inference time. `Qwen2.5-7B-Instruct` also showed competitive performance, reinforcing its consistency across different datasets. On the contrary, smaller models (e.g., `Llama-3.2-1B-Instruct`, `Llama-3.2-3B-Instruct`) achieved poor results, confirming their difficulty in predicting complex process execution dynamics. Notably, the `DeepSeek` models underperformed both in terms of predictive metrics and efficiency, showing that "reasoning" tokens in the case of this log could be detrimental.

Table 6 reports the results of the evaluation conducted on the *BPI Challenge 2013 (Incidents)* event log. Also for this log, the best-performing model across all metrics is `gpt-4o-mini`, consistently outperforming all open-source LLMs. The `phi-4` model confirms itself as the most effective open-source alternative within the framework, although at a higher computational cost (except when compared to the `DeepSeek` models, which demand even more time without delivering comparable performance). The overall average performance on this log is slightly lower compared to others, likely because, as with the *BPI Challenge 2013 (Closed Problems)* log, it contains multiple incident cases processed within each execution trace and the prediction must take into account not only the activity name but also the associated resource and lifecycle transition to better disambiguate the next event for prediction, increasing the complexity of the task for the LLMs.

Table 7 reports the performance of the framework relying on different LLMs in the next activity prediction task for *BPI Challenge 2020 (International Declarations)*. The best-performing model across all metrics is `gpt-4o-mini`, which achieves near-perfect scores (accuracy, precision, recall, and F1-score all at or above 0.98) while also maintaining a relatively low average inference time. This confirms the strength of commercial LLMs in the prediction tasks, though their closed nature may limit transparency and replicability. It is also plausible that these results are partially influenced by the model's exposure to this event log in its pre-training dataset, as the BPI Challenge datasets are well-known and widely used in the process mining community. Among the open-source models, `phi-4` delivers the most competitive performance, with all metrics above 0.93, though at a slightly higher per-inference time. `Mistral-7B-Instruct-v0.2` and `Qwen2.5-7B-Instruct` also show strong results with accuracy above 0.75 and balanced macro scores, with moderate inference times, representing good trade-offs between effectiveness and efficiency. Conversely, smaller models such as `Llama-3.2-1B-Instruct` and `Llama-3.2-3B-Instruct` perform significantly worse across all metrics, confirming that lower-capacity models struggle with the complexity of real-world process logs. Notably, while `DeepSeek-R1-Distill-Llama-8B` achieves acceptable accuracy, it exhibits the highest per-inference time, raising concerns about its usability in time-sensitive applications.

Table 6
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *BPI Challenge 2013 (Incidents)* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.37	0.33	0.32	0.32	5.28	63.3
Llama-3.2-1B-Instruct	0.03	0.03	0.02	0.03	2.65	31.8
Llama-3.2-3B-Instruct	0.05	0.06	0.05	0.05	3.05	36.5
Mistral-7B-Instruct-v0.2	0.38	0.31	0.30	0.30	4.49	53.8
Mistral-7B-Instruct-v0.3	0.12	0.12	0.11	0.11	4.38	52.5
Qwen2.5-7B-Instruct	0.45	0.32	0.32	0.31	2.71	32.5
phi-4	*0.53*	*0.57*	*0.53*	*0.54*	5.15	61.7
DeepSeek-R1-Distill-Qwen-7B	0.39	0.38	0.37	0.37	5.69	68.2
DeepSeek-R1-Distill-Llama-8B	0.46	0.43	0.42	0.43	9.52	114.2
gpt-4o-mini	**0.70**	**0.64**	**0.65**	**0.64**	1.79	21.5

Table 7
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *BPI Challenge 2020 (International Declarations)* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.71	0.74	0.68	0.67	2.65	31.8
Llama-3.2-1B-Instruct	0.23	0.31	0.18	0.19	0.71	8.5
Llama-3.2-3B-Instruct	0.39	0.61	0.42	0.48	2.05	24.5
Mistral-7B-Instruct-v0.2	0.79	0.74	0.71	0.71	2.34	28.1
Mistral-7B-Instruct-v0.3	0.63	0.50	0.47	0.46	1.29	15.4
Qwen2.5-7B-Instruct	0.75	0.82	0.80	0.78	2.13	25.5
phi-4	*0.93*	*0.94*	*0.93*	*0.93*	3.12	37.3
DeepSeek-R1-Distill-Qwen-7B	0.69	0.59	0.58	0.55	3.93	47.2
DeepSeek-R1-Distill-Llama-8B	0.70	0.70	0.69	0.67	7.34	88.1
gpt-4o-mini	**0.98**	**0.99**	**0.99**	**0.99**	0.92	11.0

Table 8
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *Hospital Billing* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.55	0.64	0.63	0.59	2.73	32.7
Llama-3.2-1B-Instruct	0.12	0.31	0.09	0.12	0.86	10.3
Llama-3.2-3B-Instruct	0.15	0.31	0.16	0.19	2.26	27.1
Mistral-7B-Instruct-v0.2	0.51	0.55	0.48	0.50	3.16	37.9
Mistral-7B-Instruct-v0.3	0.60	0.63	0.61	0.61	2.28	27.4
Qwen2.5-7B-Instruct	0.69	**0.72**	*0.74*	**0.72**	2.31	27.6
phi-4	*0.71*	0.70	**0.78**	*0.71*	3.85	46.1
DeepSeek-R1-Distill-Qwen-7B	0.63	0.68	0.68	0.67	4.58	54.9
DeepSeek-R1-Distill-Llama-8B	0.56	0.55	0.58	0.53	9.58	114.9
gpt-4o-mini	**0.73**	*0.71*	*0.74*	*0.71*	1.19	14.2

The results in Table 8 present the framework's performance for the *Hospital Billing* event log. Also for this other real-world log, `gpt-4o-mini` achieves the highest accuracy and good precision, recall, and F1-score, with `Phi-4` following closely with comparable macro metrics but reporting a higher execution time. Smaller models, such as `Llama-3.2-1B` and `Llama-3.2-3B` significantly underperform other LLMs, confirming that model size and architecture influence predictive power in this hospital domain. The `DeepSeek-R1-Distill-Qwen-7B` model performs moderately well, yet not justifying its higher resource demands due to their reasoning process. For this dataset, macro precision, recall, and F1-score closely align with accuracy, indicating a more balanced class distribution in the predictions.

Table 9 shows the outcomes of the next activity prediction task for the *Sepsis Cases* log. Here, the `Phi-4` model demonstrates the highest accuracy and balanced precision, recall, and F1-score in a time slightly higher than the average of the others, while `gpt-4o-mini` reports the highest macro metrics, also being one of the faster models in this case. Regarding the remaining models, `Qwen2.5-7B` also demonstrates acceptable performance on this log, while other open-source models do not exhibit the same level of reliability as these two. Notably, `DeepSeek` models show weaker performance, suggesting limited generalization capabilities for this task. Moreover, their results do not justify their higher resource consumption and associated costs. `Llama-3.2-1B` significantly underperforms in all the metrics, reinforcing the importance of model size and training data quality for handling event sequences in the medical domain. The close alignment between accuracy, precision, recall, and F1-score in the top-performing models suggests a more balanced class distribution in predictions compared to other datasets.

Shifting now the focus to synthetic event logs, Table 10 highlights significant discrepancies between accuracy and macro-averaged metrics in the next activity prediction task for the *SL2-2v-1r* log. In this case, all the models seem to have comparable execution times, yet they demonstrate marked differences in performance between each other and even across the various metrics. For instance, `Qwen2.5-7B-Instruct` achieves the highest accuracy, yet its macro F1-score remains low.

Table 9
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *Sepsis Cases* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.53	0.60	0.60	0.58	2.98	35.7
Llama-3.2-1B-Instruct	0.18	0.39	0.23	0.24	1.05	12.5
Llama-3.2-3B-Instruct	0.26	0.41	0.27	0.31	2.41	28.8
Mistral-7B-Instruct-v0.2	0.50	0.66	0.59	0.61	2.74	32.8
Mistral-7B-Instruct-v0.3	0.55	0.67	0.68	0.64	0.71	8.54
Qwen2.5-7B-Instruct	0.66	*0.80*	*0.79*	*0.79*	2.87	34.4
phi-4	**0.70**	0.77	0.76	0.77	3.47	41.6
DeepSeek-R1-Distill-Qwen-7B	0.55	0.56	0.60	0.54	6.86	82.3
DeepSeek-R1-Distill-Llama-8B	0.54	0.54	0.61	0.55	13.14	157.6
gpt-4o-mini	*0.69*	**0.85**	**0.83**	**0.84**	0.90	10.8

Table 10
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *SL2-2v-1r* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.39	*0.17*	*0.16*	*0.16*	0.33	25.8
Llama-3.2-1B-Instruct	0.09	0.14	0.02	0.03	0.17	13.3
Llama-3.2-3B-Instruct	0.13	0.14	0.06	0.08	0.24	18.8
Mistral-7B-Instruct-v0.2	0.18	0.10	0.09	0.08	0.50	39.1
Mistral-7B-Instruct-v0.3	0.20	0.16	0.11	0.11	0.33	28.8
Qwen2.5-7B-Instruct	**0.81**	**0.25**	**0.22**	**0.22**	0.39	30.5
phi-4	*0.72*	0.14	0.14	0.13	0.67	52.43
DeepSeek-R1-Distill-Qwen-7B	0.69	0.14	0.13	0.13	0.50	39.1
DeepSeek-R1-Distill-Llama-8B	0.58	0.13	0.10	0.11	1.34	104.8
gpt-4o-mini	0.70	0.14	0.14	0.14	0.13	10.4

Table 11
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *SL3-multivariant* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.25	0.10	0.09	0.09	2.40	28.8
Llama-3.2-1B-Instruct	0.07	0.07	0.01	0.02	0.82	9.8
Llama-3.2-3B-Instruct	0.42	0.06	0.06	0.06	1.89	22.6
Mistral-7B-Instruct-v0.2	0.23	0.11	0.07	0.08	3.44	41.2
Mistral-7B-Instruct-v0.3	0.22	0.11	0.08	0.09	2.82	33.8
Qwen2.5-7B-Instruct	*0.46*	**0.14**	**0.14**	**0.14**	2.44	29.2
phi-4	0.42	0.12	0.10	0.11	3.83	46.0
DeepSeek-R1-Distill-Qwen-7B	0.42	0.12	0.10	0.11	2.94	35.3
DeepSeek-R1-Distill-Llama-8B	0.40	0.08	0.06	0.06	10.59	127.1
gpt-4o-mini	**0.47**	*0.13*	*0.13*	*0.13*	1.04	12.4

Similarly, `Phi-4` and `DeepSeek-R1-Distill-Qwen-7B` exhibit good accuracy but low macro F1-scores. This suggests that while these models perform well on frequently occurring classes of activities, they struggle with less frequent ones, leading to poor overall balance across classes. This phenomenon may be attributed to an imbalance in the event log, likely resulting from its synthetic generation. Smaller models such as `Llama-3.2-1B-Instruct` and `Llama-3.2-3B-Instruct` again perform poorly across all metrics, reaffirming that a higher model dimension is fundamental also for predicting activities in synthetic logs effectively.

The outcomes in Table 11 also report important variations in model performance for the *SL3-multivariant* log, both in execution time and performance. Overall, this log highlights a significant downgrade across all the metrics for the framework, potentially due to the extremely unpredictable nature intentionally imposed by design on its traces. `gpt-4o-mini` achieves the highest accuracy, followed by `Qwen2.5-7B-Instruct`, indicating its predictive capabilities on frequently occurring activities. However, the macro-averaged precision, recall, and F1-score remain consistently low across all models, suggesting challenges in correctly predicting less frequent activities. These results suggest that while some models perform well in overall accuracy, their predictions may be biased toward common activity classes, failing to generalize effectively to rarer events. Again, the smaller model, i.e., `Llama-3.2-1B-Instruct`, is the worst performing one.

Table 12 shows the results for the *Udon-ya* log. Among the open-source models, `Qwen2.5-7B-Instruct` performs best across all metrics, achieving an accuracy of 0.72 and an F1-score of 0.70, confirming its strong generalization even in semi-structured domains. `Phi-4` follows closely, though with slightly lower precision and recall. Notably, the closed-source `gpt-4o-mini` outperforms all others, achieving the highest accuracy and macro-average scores, suggesting it benefits from a broader and possibly from the exposition to this domain in its training corpus.

On the other hand, smaller models like `Llama-3.2-1B-Instruct` perform poorly, highlighting the limitations of limited-capacity LLMs on complex real-world data. The execution

Table 12
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score in the next activity prediction task for the *Udon-ya* log. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Language model	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
Llama-3.1-8B-Instruct	0.56	0.57	0.51	0.53	3.00	36.0
Llama-3.2-1B-Instruct	0.14	0.27	0.11	0.11	0.71	8.4
Llama-3.2-3B-Instruct	0.38	0.48	0.30	0.36	2.36	28.3
Mistral-7B-Instruct-v0.2	0.54	0.54	0.45	0.47	2.39	28.6
Mistral-7B-Instruct-v0.3	0.63	0.54	0.50	0.50	1.16	13.9
Qwen2.5-7B-Instruct	*0.72*	*0.75*	*0.68*	*0.70*	2.46	29.5
phi-4	0.65	0.69	0.61	0.63	3.71	44.5
DeepSeek-R1-Distill-Qwen-7B	0.61	0.49	0.48	0.47	3.73	44.7
DeepSeek-R1-Distill-Llama-8B	0.58	0.55	0.50	0.51	6.89	82.6
gpt-4o-mini	**0.84**	**0.80**	**0.79**	**0.78**	1.37	16.4

times remain within acceptable ranges, with models like `Mistral-7B-Instruct-v0.3` and `gpt-4o-mini` demonstrating good trade-offs between performance and latency, which is a desirable characteristic for deployment in time-sensitive environments.

The gap between accuracy and macro metrics we can observe across many of these evaluations highlights the importance of using multiple balanced evaluation criteria in PPM rather than relying solely on accuracy.

***Rq2**: is the approach effective compared to the baseline?*

To investigate **RQ2**, we analyzed the performance of several state-of-the-art approaches from the literature on the real-world event logs considered in this study using our proposed framework. Table 13 presents this comparison based on accuracy in the next activity prediction task.

Concerning the *BPI Challenge 2012* event log, our approach slightly underperforms the best-performing techniques, likely due to a *lost-in-the-middle* phenomenon originating from the length of the retrieved prefixes with attributes to load in the LLM context [50].

Conversely, for the *BPI Challenge 2013* dataset, the proposed approach, when using `gpt-4o-mini`, achieves significantly better accuracy than state-of-the-art baselines on the *Closed Problems* log. On the *Incidents* log, it performs above average and closely follows the results of the best existing techniques.

For the *BPI Challenge 2020 (International Declarations)* event log, our approach achieves an accuracy slightly lower than the only existing study using the same log for next activity prediction [74].

Our approach on the *Hospital Billing* log closely matches the best-performing method [73] in accuracy. Notably, it outperforms other deep learning-based solutions, including [3,71], and [40], which achieve lower scores. This result indicates that our model is highly competitive in structured business processes, effectively capturing process dynamics despite the presence of categorical attributes and complex dependencies.

The *Sepsis Cases* dataset represents a particularly challenging scenario, as evidenced by the generally lower performance of several baseline models. Our approach significantly outperforms all existing methods, achieving the best results among the compared techniques. In contrast, previous deep learning-based methods, such as LSTM-based models [3], encoder–decoder approaches [72], and GAN-based models [75], exhibit considerably lower accuracy. This substantial improvement suggests that our approach effectively captures the complex sequential dependencies and semantic aspects inherent in medical event logs. Compared to SNAP [11], which also leverages LLMs for predictions, our method demonstrates significant gains, highlighting the benefits of incorporating RAG for a more refined and adaptive prediction model.

Overall, our results show that the proposed approach achieves performance comparable to the top state-of-the-art techniques while requiring no additional training, unlike other deep learning-based methods, which typically involve extensive model training and fine-tuning. This highlights its efficiency and adaptability across different process domains.

***Rq3**: how robust is the performance of the llms embedded in the proposed approach?*

Robustness analysis. To address **RQ3**, we investigated the robustness of the LLMs embedded into our framework by analyzing the distribution of their prediction accuracy. Fig. 5 presents a box plot reporting the variance in accuracy between the evaluations on different event logs, providing insights into the reliability of each model.

Notably, `Qwen2.5-7B-Instruct` and `Phi-4` achieve the highest median accuracy among the open-source models, with relatively small variance ranges. The limited presence of outliers further indicates that these models maintain stable prediction performance across different event logs. Conversely, models smaller in size, such as `Llama-3.2-1B-Instruct` and `Llama-3.2-3B-Instruct`, exhibit not only lower median accuracy but also wider ranges, demonstrating higher variability and reduced robustness.

Another observation we can derive from the box blot is on the performance of `Mistral-7B-Instruct-v0.2` and `Mistral-7B-Instruct-v0.3`. While both models achieve acceptable accuracy, their larger variance suggests that their prediction quality is more sensitive to the input data. This may indicate challenges in generalizing to different cases within the event log.

Furthermore, the proprietary `gpt-4o-mini` presents a relatively compact range with the best median accuracy, reflecting a balance between stability and performance. Similarly, the `DeepSeek-R1-Distill` models exhibit mixed behaviors. While `DeepSeek-R1-Distill-Qwen-7B` achieves a higher maximum accuracy, `DeepSeek-R1-Distill-Llama-8B` shows a narrower interquartile range of accuracy values, indicating greater robustness.

Posthoc statistical tests. Still tackling **RQ3**, we applied a *Friedman test* followed by a *Nemenyi test* to determine whether the differences in performance among the evaluated LLMs are statistically significant. The Friedman test, applied across all models using their mean scores on the various event logs, returned a value of 25.56 with a *p-value* of 1.18×10^{-5}, indicating that the observed differences in rankings across models are highly significant and unlikely due to random chance. Thus, we can reject the null hypothesis H_0.

To investigate further which model differences were statistically meaningful, we applied the Nemenyi test. The results, reported in Table 14 for the three best LLMs, show that `gpt-4o-mini` significantly outperformed open-source models, with *p-values* even below 0.01. The differences between some of the top-performing open-source models, such as `phi-4` and `Qwen2.5-7B-Instruct`, were less pronounced but still statistically meaningful. Interestingly, the test also confirmed the lack of significant difference between some models (e.g., `Mistral-7B-v0.2` and the `DeepSeek` variants), further validating the trends

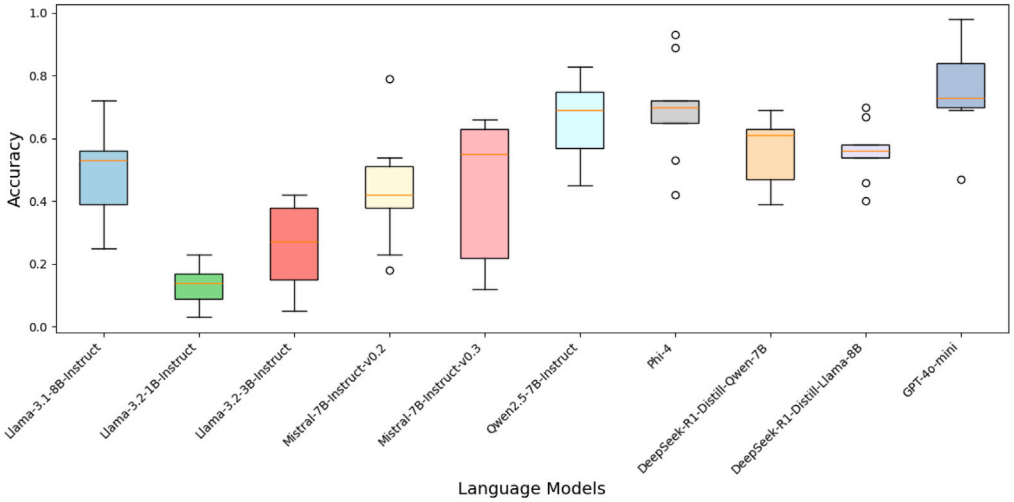


Fig. 5. Box plot illustrating the range of accuracy scores obtained throughout the framework evaluation by the language models across the considered event logs.

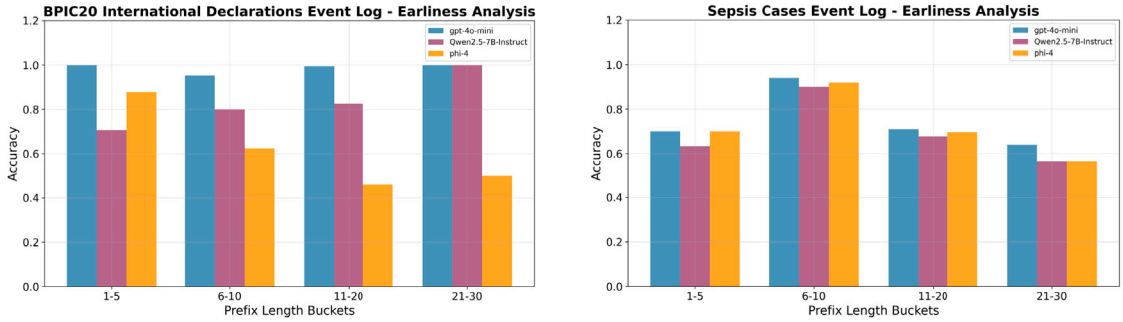


Fig. 6. Results of the earliness analysis on the BPI Challenge 2020 International Declarations (left) and Sepsis Cases (right) event logs.

observed in the previous evaluations. These findings support the claim that `gpt-4o-mini` remains the best model to work with the proposed framework, while `phi-4` and `Qwen2.5-7B-Instruct` lead among open-source ones.

Earliness analysis. Subsequently, we performed the analysis of *earliness* for the top three models (i.e., `gpt-4o-mini`, `Qwen2.5-7B-Instruct`, and `phi-4`) over the *BPI Challenge 2020 International Declarations* and *Sepsis Cases* logs. The results are presented in Fig. 6.

For the *BPI Challenge 2020 International Declarations* log (left), `gpt-4o-mini` consistently achieves high accuracy across all prefix ranges, confirming its robustness in both early and late stages of process execution. Among open-source models, `Qwen2.5-7B-Instruct` demonstrates a gradual improvement in accuracy as the prefix length increases, demonstrating the capability to manipulate richer context effectively and leverage more information from longer prefixes. In contrast, `phi-4` shows strong early performance but exhibits a noticeable decline for predictions over longer prefixes, suggesting a drop in generalization as the trace length increases.

On the *Sepsis Cases* log (right), the models exhibit more variability but show a consistent trend. `gpt-4o-mini` better performs in accuracy at mid-length prefixes (i.e., from 6 to 10 activities), but performance slightly declines for longer ones. `Qwen2.5-7B-Instruct` and `phi-4` follow a similar pattern, registering the highest accuracy for the same bucket, while progressively decreasing afterwards. This behavior is likely due to the increased complexity and variability of the *Sepsis Cases* log, where longer prefixes might include more divergence in the process flow, making the prediction task more challenging.

These findings highlight how early-stage prediction is feasible and often effective, while also showcasing how both LLM's capability and log complexity influence performance.

Model fine-tuning. Table 15 reports the evaluation results of the framework using the fine-tuned `phi-4` model adapted via LoRA over the *S_NAP* dataset, following the same settings used in [7]. The fine-tuned version exhibits a notable decline in predictive performance across all event logs compared to the original `phi-4` model. For instance, on the *BPI Challenge 2020 (International Declarations)* log, the regular `phi-4` achieved an accuracy of 0.93 and a (macro) F1-score of 0.93, while the fine-tuned variant only registered an accuracy of 0.39 and (macro) F1 of 0.34. Similarly, decreases are evident across other logs, such as the *Udon-ya* log (in which the accuracy is one-third of that for the original LLM) and the *Sepsis Cases* log (where the accuracy is halved).

Despite this degradation in effectiveness, a key benefit of the fine-tuned model is its improved computational efficiency since, on average, the inference time is reduced by almost half. For instance, in the *BPI Challenge 2020* log, the per-inference time drops from 37.3 s to 19.6 s for the fine-tuned `phi-4`. This trade-off suggests that the fine-tuned model, although tailored on a dataset specific for the next activity prediction task, may suffer from overfitting to the considered domains and lose the generalization ability on new event logs, not justifying the significant computational costs required by the fine-tuning process. Moreover, this performance degradation may also originate from the fact that the fine-tuning process, as performed on the *S_NAP* dataset, does not teach the model how to effectively exploit the retrieved traces and their data attributes values for the prediction of the next event.

Hence, such an investigation underlines the importance of the dataset selection and fine-tuning strategies when adapting LLMs for PPM. While fine-tuning can offer efficiency improvements, it may require further adaptations or data augmentation to maintain the generalization capabilities on new datasets.

Table 13
Comparison with state-of-the-art approaches on the real-world event logs in terms of accuracy, primarily based on [11], [68], [69], and [70]. The **best** and *second-best* results for each metric are reported in **bold** and *italics*, respectively.

Dataset	Study	Accuracy
BPI Challenge 2012	[3]	*0.85*
	[36]	0.59
	[40]	0.83
	[42]	0.83
	[69]	**0.86**
	[70]	0.78
	[71]	**0.86**
	[41]	0.84
	[72]	0.42
	[35] (w/o attributes)	0.82
	[35] (w/ attributes)	0.80
	[73]	0.85
	Our approach	0.76
BPI Challenge 2013 (Closed Problems)	[3]	0.64
	[11]	*0.69*
	[36]	0.58
	[40]	0.24
	[42]	0.54
	[71]	0.63
	[41]	0.24
	[72]	0.43
	[35] (w/o attributes)	0.59
	[35] (w/ attributes)	0.54
	Our approach	**0.90**
BPI Challenge 2013 (Incidents)	[3]	0.70
	[11]	0.67
	[36]	0.66
	[40]	0.46
	[42]	0.66
	[69]	**0.79**
	[71]	*0.74*
	[41]	0.36
	[72]	0.51
	[35] (w/o attributes)	0.59
	[35] (w/ attributes)	0.51
	[73]	0.62
	Our approach	0.70
BPI Challenge 2020 (Int. Decl.)	[74]	*0.84*
	Our approach	**0.98**
Hospital Billing	[3]	0.78
	[40]	0.75
	[69]	*0.84*
	[71]	0.79
	[72]	0.70
	[73]	**0.85**
	Our approach	0.73
Sepsis Cases	[3]	0.64
	[11]	*0.66*
	[36]	0.40
	[40]	0.56
	[69]	0.62
	[70]	0.65
	[71]	0.63
	[72]	0.21
	[35] (w/o attributes)	0.55
	[35] (w/ attributes)	0.58
	[75]	0.60
	Our approach	**0.70**

6. Threats to validity

In this section, we discuss potential threats to the validity of the study. An internal validity concern refers to the rapid advancement of LLMs (new versions are developed in a few months) that potentially makes the obtained results outdated over time. To address this, we evaluated numerous state-of-the-art models, both open-source and proprietary, demonstrating the framework's versatility and independence from specific technologies.

Instead, an external validity threat to this work is given by the difficulty in replicating the proposed experiments due to the statistical variability in LLM outputs even in response to identical prompts. To mitigate this problem, we made the employed prompts available online. This makes the experiments more reproducible and reduces the differences in how questions or commands are phrased. Moreover, we performed multiple runs along the experiment to ensure the correctness of the answer. Always referring to the external validity, we also focused on ensuring the generalizability of our results by extending the experimentation to a larger number of event logs, both real-world and synthetic ones, in a wide ranging of domains.

7. Conclusion

This paper introduced a novel approach for next activity prediction in PPM, leveraging RAG to enhance the predictive capabilities of LLMs. Our experiments demonstrated that LLMs, when solely relying on task instructions and minimal examples, struggle to perform complex process mining tasks. However, integrating RAG to provide additional contextual information about the traces and their data attributes from event logs, along with carefully designed prompt engineering, significantly improves predictive accuracy and robustness.

The evaluation across multiple (real-world and synthetic) event logs and experimental configurations showed that our approach achieves competitive performance compared to state-of-the-art deep learning models. Notably, while deep learning-based techniques require extensive training and feature engineering, our method operates without fine-tuning, making it highly adaptable across different domains. Additionally, by systematically assessing multiple LLMs, we highlighted variations in predictive performance, demonstrating that some models exhibit greater robustness across diverse process domains.

Moreover, we reinforced this evaluation by confirming the statistical significance of the results through the Friedman and Nemenyi tests. We further analyzed the early prediction capabilities of the top-performing models, showing that our framework can maintain high accuracy even with limited prefix information, which is a fundamental requirement for real-world PPM. Finally, for completeness, we explored fine-tuning for improving the LLM's performance on this prediction task. Although fine-tuning reduced inference time considerably, it often led to a trade-off in prediction quality, suggesting that it may not always be beneficial in PPM scenarios.

Despite these strengths, the study presents some limitations. The framework's effectiveness diminishes in logs with frequent interleaving activities, especially when considering full activity lifecycle stages (e.g., assign, start, complete). Furthermore, our approach does not explicitly address concept drift, which can impact performance when process behaviors evolve over time. Future research could investigate strategies for dynamically adapting LLM predictions to drift patterns and explore ensemble methods to mitigate variability in the language model performance.

Moreover, the experiments emphasized the need for standardized benchmarks for LLM-based activity prediction, ensuring more comprehensive and reproducible evaluations in future studies. While accuracy serves as a key performance metric, our analysis demonstrated the importance of macro-aggregated precision, recall, and F1-score in understanding model robustness across different activity classes. Addressing class imbalance through data augmentation or targeted fine-tuning strategies could further enhance predictive stability.

Finally, another extension to explore in future work will focus on optimizing the computational efficiency and reducing the inference latency of the framework. While our current approach demonstrates competitive performance, its applicability in resource-constrained or time-sensitive operational settings could be enhanced by integrating

Table 14
Extract of Nemenyi test results for the three best models, indicating the *p-values* comparing each pair of LLMs (row vs. column). Lower values report stronger evidence against the null hypothesis H_0 of equal performance.

	gpt-4o-mini	*phi-4*	*Qwen2.5-7B-Instruct*
gpt-4o-mini	1.000	0.726	0.002
phi-4	0.726	1.000	0.046
Qwen2.5-7B-Instruct	0.002	0.046	1.000

Table 15
Accuracy, (macro) precision, (macro) recall, and (macro) F1-score achieved by the framework with the fine-tuned `phi-4` model in the next activity prediction task across all the considered event logs. The execution time is reported as total time (in *hours*) and average per-inference time (in *seconds*).

Event Log	Accuracy	Precision	Recall	F1-score	Time	
					Total (h)	*Per-Inference (s)*
BPIC 2012	0.37	0.18	0.26	0.19	2.06	24.8
BPIC 2013 (Closed Problems)	0.11	0.10	0.10	0.10	0.69	8.2
BPIC 2013 (Incidents)	0.27	0.29	0.27	0.27	2.06	24.7
BPIC 2020 (Int. Decl.)	0.39	0.35	0.38	0.34	1.64	19.6
Hospital Billing	0.44	0.49	0.39	0.42	1.53	18.3
Sepsis Cases	0.39	0.28	0.28	0.27	1.85	22.2
SL2-2v-1r	0.35	0.16	0.10	0.11	0.25	19.7
SL3-multivariant	0.49	0.13	0.08	0.10	2.16	25.8
Udon-ya	0.24	0.29	0.28	0.28	2.47	29.6

more efficient RAG architectures. For instance, exploring variants like FastRAG [76], which can decouple the retrieval and generation steps to reduce computational overhead without significantly compromising retrieval quality, presents a promising avenue. Adopting such architectures could make retrieval-augmented LLMs more viable for real-time PPM, where accurate and rapid forecasts are essential for decision-making.

CRediT authorship contribution statement

Angelo Casciani: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Formal analysis, Data curation, Conceptualization. **Mario Luca Bernardi:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Formal analysis, Conceptualization. **Marta Cimitile:** Writing – review & editing, Writing – original draft, Supervision, Methodology, Formal analysis, Conceptualization. **Andrea Marrella:** Writing – review & editing, Supervision, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

We acknowledge financial support under the National Recovery and Resilience Plan (NRRP), M4C2I1.1, funded by the European Union – NextGenerationEU – Project Title aRtificial intElligence for Process Analytics (REPA) - Grant Assignment Decree No. 2022CJWPNA by the Italian Ministry of University and Research (MUR), the Sapienza project FOND-AIBPM, Italy, the PRIN 2022 project MOTOWN, and the NRRP MUR project PE0000013-FAIR, Italy. This work has been carried out while Angelo Casciani was enrolled in the Italian National Doctorate on Artificial Intelligence run by Sapienza University of Rome (CUP B53C23003500006, Mission 4 NRRP Generic Research Scholarship, Component 1).

Data availability

The source code, event logs, prompts, and instructions for replicating the experiments are available at: https://github.com/angelo-casciani/rag_next_activity. We also make the fine-tuned language model available online for testing at: https://huggingface.co/angeloc1/autotrain-phi-ft-nap.

References

[1] W.M.P. van der Aalst, Process Mining - Data Science in Action, second ed., Springer, 2016, http://dx.doi.org/10.1007/978-3-662-49851-4.

[2] C. Di Francescomarino, C. Ghidini, Predictive process monitoring, in: Process Mining Handbook, in: Lecture Notes in Business Information Processing, vol. 448, Springer, 2022, pp. 320–346, http://dx.doi.org/10.1007/978-3-031-08848-3_10.

[3] N. Tax, I. Verenich, M.L. Rosa, M. Dumas, Predictive business process monitoring with LSTM neural networks, in: E. Dubois, K. Pohl (Eds.), Advanced Information Systems Engineering - 29th International Conference, CAiSE 2017, Essen, Germany, June (2017) 12-16, Proceedings, in: Lecture Notes in Computer Science, vol. 10253, Springer, 2017, pp. 477–492, http://dx.doi.org/10.1007/978-3-319-59536-8_30.

[4] F.M. Maggi, C. Di Francescomarino, M. Dumas, C. Ghidini, Predictive monitoring of business processes, in: Advanced Information Systems Engineering - 26th International Conference, CAiSE 2014, Thessaloniki, Greece, June (2014) 16-20. Proceedings, in: Lecture Notes in Computer Science, vol. 8484, Springer, 2014, pp. 457–472, http://dx.doi.org/10.1007/978-3-319-07881-6_31.

[5] L.T. Ly, F.M. Maggi, M. Montali, S. Rinderle-Ma, W.M.P. van der Aalst, Compliance monitoring in business processes: Functionalities, application, and tool-support, Inf. Syst. 54 (2015) 209–234, http://dx.doi.org/10.1016/J.IS.2015.02.007.

[6] B. Estrada-Torres, A. del-Río-Ortega, M. Resinas, Mapping the landscape: Exploring large language model applications in business process management, in: Enterprise, Business-Process and Information Systems Modeling - 25th International Conference, BPMDS 2024, and 29th International Conference, EMMSAD 2024, Limassol, Cyprus, June (2024) 3-4, Proceedings, in: Lecture Notes in Business Information Processing, vol. 511, Springer, 2024, pp. 22–31, http://dx.doi.org/10.1007/978-3-031-61007-3_3.

[7] A. Rebmann, F.D. Schmidt, G. Glavas, H. van der Aa, Evaluating the ability of llms to solve semantics-aware process mining tasks, in: 6th International Conference on Process Mining, ICPM 2024, Kgs. Lyngby, Denmark, October 2024 14-18, IEEE, 2024, pp. 9–16, http://dx.doi.org/10.1109/ICPM63005.2024.10680677.

[8] M.L. Bernardi, A. Casciani, M. Cimitile, A. Marrella, Conversing with business process-aware large language models: the BPLLM framework, J. Intell. Inf. Syst. 62 (6) (2024) 1607–1629, http://dx.doi.org/10.1007/S10844-024-00898-1.

[9] C. Di Francescomarino, C. Ghidini, F.M. Maggi, F. Milani, Predictive process monitoring methods: Which one suits me best? in: M. Weske, M. Montali, I. Weber, J. vom Brocke (Eds.), Business Process Management - 16th International

Conference, BPM 2018, Sydney, NSW, Australia, September (2018) 9-14, Proceedings, in: Lecture Notes in Computer Science, vol. 11080, Springer, 2018, pp. 462–479, http://dx.doi.org/10.1007/978-3-319-98648-7_27.

[10] R. Aringhieri, G. Boella, E. Brunetti, L.D. Caro, et al., Leveraging structured data in predictive process monitoring: the case of the ICD-9-CM in the scenario of the home hospitalization service, in: Proceedings of the Workshop on Towards Smarter Health Care: Can Artificial Intelligence Help? Co-Located with 20th International Conference of the Italian Association for Artificial Intelligence (AIxIA2021), Anywhere, November 29th, 2021 3060of CEUR Workshop Proceedings, 2021, pp. 48–60, CEUR-WS.org, https://ceur-ws.org/Vol-3060/paper-6.pdf.

[11] A. Oved, S. Shlomov, S. Zeltyn, N. Mashkif, A. Yaeli, SNAP: semantic stories for next activity prediction, in: T. Walsh, J. Shah, Z. Kolter (Eds.), AAAI-25, Sponsored By the Association for the Advancement of Artificial Intelligence, February 25 - March 4 2025, Philadelphia, PA, USA, AAAI Press, 2025, pp. 28871–28877, http://dx.doi.org/10.1609/AAAI.V39I28.35153.

[12] R. Galanti, B. Coma-Puig, M. de Leoni, J. Carmona, N. Navarin, Explainable predictive process monitoring, in: B.F. van Dongen, M. Montali, M.T. Wynn (Eds.), 2nd International Conference on Process Mining, IEEE, ICPM 2020, Padua, Italy, October (2020) 4-9, 2020, pp. 1–8, http://dx.doi.org/10.1109/ICPM49681.2020.00012.

[13] C. Di Francescomarino, M. Dumas, F.M. Maggi, I. Teinemaa, Clustering-based predictive process monitoring, IEEE Trans. Serv. Comput. 12 (6) (2019) 896–909, http://dx.doi.org/10.1109/TSC.2016.2645153.

[14] W.X. Zhao, K. Zhou, J. Li, T. Tang, et al., A survey of large language models, 2023, http://dx.doi.org/10.48550/ARXIV.2303.18223, CoRR abs/2303.18223, arXiv:2303.18223.

[15] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, et al., Attention is all you need, in: Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017.

[16] OpenAI, GPT-4 technical report, 2023, http://dx.doi.org/10.48550/ARXIV.2303.08774, CoRR abs/2303.08774, arXiv:2303.08774.

[17] A. Dubey, A. Jauhri, A. Pandey, A. Kadian, et al., The llama 3 herd of models, 2024, http://dx.doi.org/10.48550/ARXIV.2407.21783, CoRR abs/2407.21783, arXiv:2407.21783.

[18] A.Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, et al., Mistral 7b, 2024, http://dx.doi.org/10.48550/ARXIV.2310.06825, CoRR abs/2310.06825, arXiv:2310.06825.

[19] M. Abdin, J. Aneja, H. Behl, S. Bubeck, et al., Phi-4 technical report, 2024, http://dx.doi.org/10.48550/ARXIV.2412.08905, CoRR abs/2412.08905, arXiv:2412.08905.

[20] T. Kojima, S.S. Gu, M. Reid, Y. Matsuo, Y. Iwasawa, Large language models are zero-shot reasoners, in: Advances in Neural Information Processing Systems (NeurIPS), vol. 35, 2022, pp. 22199–22213, http://papers.nips.cc/paper_files/paper/2022/hash/8bb0d291acd4acf06ef112099c16f326-Abstract-Conference.html.

[21] N. Chirkova, S. Troshin, Empirical study of transformers for source code, in: ESEC/FSE '21: 29th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ACM, 2021, pp. 703–715, http://dx.doi.org/10.1145/3468264.3468611.

[22] D. Guo, D. Yang, H. Zhang, J. Song, et al., Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025, http://dx.doi.org/10.48550/ARXIV.2501.12948, CoRR abs/2501.12948, arXiv:2501.12948.

[23] G.E. Hinton, O. Vinyals, J. Dean, Distilling the knowledge in a neural network, 2015, CoRR abs/1503.02531, arXiv:1503.02531.

[24] V. Rawte, A. Sheth, A. Das, A survey of hallucination in large foundation models, 2024, http://dx.doi.org/10.48550/ARXIV.2309.05922, CoRR abs/2309.05922, arXiv:2309.05922.

[25] Y. Gao, Y. Xiong, X. Gao, K. Jia, et al., Retrieval-augmented generation for large language models: A survey, 2023, http://dx.doi.org/10.48550/ARXIV.2312.10997, CoRR abs/2312.10997, arXiv:2312.10997.

[26] Z. Han, C. Gao, J. Liu, J. Zhang, S.Q. Zhang, Parameter-efficient fine-tuning for large models: A comprehensive survey, 2024, http://dx.doi.org/10.48550/ARXIV.2403.14608, CoRR abs/2403.14608, arXiv:2403.14608.

[27] Q. Dong, L. Li, D. Dai, C. Zheng, et al., A survey on in-context learning, in: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, EMNLP 2024, Association for Computational Linguistics, 2024, pp. 1107–1128, https://aclanthology.org/2024.emnlp-main.64.

[28] P.S.H. Lewis, E. Perez, A. Piktus, F. Petroni, et al., Retrieval-augmented generation for knowledge-intensive NLP tasks, in: Advances in Neural Information Processing Systems 33 (NeurIPS), vol. 33, 2020, pp. 9459–9474, https://proceedings.neurips.cc/paper/2020/hash/6b493230205f780e1bc26945df7481e5-Abstract.html.

[29] B. Wang, A. Wang, F. Chen, Y. Wang, C.J. Kuo, Evaluating word embedding models: Methods and experimental results, 2019, CoRR abs/1901.09785, arXiv:1901.09785. http://arxiv.org/abs/1901.09785.

[30] T. Mikolov, K. Chen, G. Corrado, J. Dean, Efficient estimation of word representations in vector space, in: 1st International Conference on Learning Representations, ICLR 2013, Scottsdale, Arizona, USA, May (2013) 2-4, Workshop Track Proceedings, 2013, http://arxiv.org/abs/1301.3781.

[31] M.V. Koroteev, BERT: A review of applications in natural language processing and understanding, 2021, CoRR abs/2103.11943, arXiv:2103.11943. https://arxiv.org/abs/2103.11943.

[32] M. Günther, S. Sturua, M.K. Akram, I. Mohr, A. Ungureanu, B. Wang, S. Eslami, S. Martens, M. Werk, N. Wang, H. Xiao, Jina-embeddings-v4: Universal embeddings for multimodal multilingual retrieval, 2025, http://dx.doi.org/10.48550/ARXIV.2506.18902, CoRR abs/2506.18902, arXiv:2506.18902.

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

[34] A.E. Márquez-Chamorro, M. Resinas, A. Ruiz-Cortés, Predictive monitoring of business processes: A survey, IEEE Trans. Serv. Comput. 11 (6) (2018) 962–977, http://dx.doi.org/10.1109/TSC.2017.2772256.

[35] J. Theis, H. Darabi, Decay replay mining to predict next process events, IEEE Access 7 (2019) 119787–119803, http://dx.doi.org/10.1109/ACCESS.2019.2937085.

[36] J. Evermann, J. Rehse, P. Fettke, Predicting process behaviour using deep learning, Decis. Support Syst. 100 (2017) 129–140, http://dx.doi.org/10.1016/J.DSS.2017.04.003.

[37] H. Chen, X. Fang, H. Fang, Multi-task prediction method of business process based on bert and transfer learning, Knowl.-Based Syst. 254 (2022) 109603, http://dx.doi.org/10.1016/j.knosys.2022.109603.

[38] G. Park, M. Song, Prediction-based resource allocation using lstm and minimum cost and maximum flow algorithm, in: 2019 International Conference on Process Mining, ICPM, 2019, pp. 121–128, http://dx.doi.org/10.1109/ICPM.2019.00027.

[39] E. Obodoekwe, X. Fang, K. Lu, Convolutional neural networks in process mining and data analytics for prediction accuracy, Electronics 11 (14) (2022) http://dx.doi.org/10.3390/electronics11142128.

[40] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, Using convolutional neural networks for predictive process analytics, in: 2019 International Conference on Process Mining, ICPM, 2019, pp. 129–136, http://dx.doi.org/10.1109/ICPM.2019.00028.

[41] N.D. Mauro, A. Appice, T.M.A. Basile, Activity prediction of business process instances with inception CNN models, in: M. Alviano, G. Greco, F. Scarcello (Eds.), AI*IA 2019 - Advances in Artificial Intelligence - XVIIIth International Conference of the Italian Association for Artificial Intelligence, Rende, Italy, November (2019) 19-22, Proceedings, in: Lecture Notes in Computer Science, vol. 11946, Springer, 2019, pp. 348–361, http://dx.doi.org/10.1007/978-3-030-35166-3_25.

[42] M. Camargo, M. Dumas, O. González-Rojas, Learning accurate lstm models of business processes, in: Business Process Management: 17th International Conference, BPM 2019, Vienna, Austria, September (2019) 1–6, Proceedings, Springer-Verlag, Berlin, Heidelberg, 2019, pp. 286–302, http://dx.doi.org/10.1007/978-3-030-26619-6_19.

[43] L. Lin, L. Wen, J. Wang, Mm-pred: A deep predictive model for multi-attribute event sequence, in: Proceedings of the 2019 SIAM International Conference on Data Mining, SDM 2019, Calgary, Alberta, Canada, May (2019) 2-4, SIAM, 2019, pp. 118–126, http://dx.doi.org/10.1137/1.9781611975673.14.

[44] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, Leveraging multi-view deep learning for next activity prediction, in: A. Marrella, D.T. Dupré (Eds.), Proceedings of the 1st Italian Forum on Business Process Management Co-Located with the 19th International Conference of Business Process Management (BPM 2021), Rome, Italy, September 10th, 2021 2952of CEUR Workshop Proceedings, 2021, pp. 1–6, CEUR-WS.org, http://ceur-ws.org/Vol-2952/paper_290a.pdf.

[45] F. Taymouri, M.L. Rosa, S.M. Erfani, Z.D. Bozorgi, I. Verenich, Predictive business process monitoring via generative adversarial nets: The case of next event prediction, 2020, CoRR abs/2003.11268, arXiv:2003.11268.

[46] V. Pasquadibisceglie, A. Appice, D. Malerba, Lupin: A llm approach for activity suffix prediction in business process event logs, in: 2024 6th International Conference on Process Mining, ICPM, 2024, pp. 1–8, http://dx.doi.org/10.1109/ICPM63005.2024.10680620.

[47] V. Pasquadibisceglie, A. Appice, G. Castellano, D. Malerba, Enhancing next activity prediction with adversarial training of vision transformers, in: Proceedings of the 32nd Symposium of Advanced Database Systems, Villasimius, Italy, June 23rd to 26th, 2024 3741 of CEUR Workshop Proceedings, 2024, pp. 349–358, CEUR-WS.org, https://ceur-ws.org/Vol-3741/paper17.pdf.

[48] P. Ceravolo, M. Comuzzi, J. De Weerdt, C. Di Francescomarino, F.M. Maggi, Predictive process monitoring: concepts, challenges, and future research directions, Process. Sci. 1 (1) (2024) 1–22.

[49] G. Acampora, A. Vitiello, et al., IEEE 1849: The XES standard, IEEE Comput. Intell. Mag. 12 (2) (2017) 4–8.

[50] N.F. Liu, K. Lin, J. Hewitt, A. Paranjape, M. Bevilacqua, F. Petroni, P. Liang, Lost in the middle: How language models use long contexts, Trans. Assoc. Comput. Linguist. 12 (2024) 157–173, http://dx.doi.org/10.1162/TACL_A_00638.

[51] J. White, Q. Fu, S. Hays, M. Sandborn, et al., A prompt pattern catalog to enhance prompt engineering with chatgpt, 2023, http://dx.doi.org/10.48550/ARXIV.2302.11382, CoRR abs/2302.11382, arXiv:2302.11382.

[52] M. Shanahan, K. McDonell, L. Reynolds, Role play with large language models, Nature 623 (7987) (2023) 493–498, http://dx.doi.org/10.1038/S41586-023-06647-8.

[53] A. Parnami, M. Lee, Learning from few examples: A summary of approaches to few-shot learning, 2022, http://dx.doi.org/10.48550/ARXIV.2203.04291, CoRR abs/2203.04291, arXiv:2203.04291.

[54] B. van Dongen, Bpi challenge 2012, 2012, http://dx.doi.org/10.4121/uuid:3926db30-f712-4394-aebc-75976070e91f, https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204/1.

[55] W. Steeman, Bpi challenge 2013, closed problems, 2013, http://dx.doi.org/10.4121/uuid:c2c3b154-ab26-4b31-a0e8-8f2350ddac11, https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_closed_problems/12714476/1.

[56] W. Steeman, Bpi challenge 2013, incidents, 2013, http://dx.doi.org/10.4121/uuid:500573e6-accc-4b0c-9576-aa5468b10cee, https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_incidents/12693914/1.

[57] B. van Dongen, Bpi Challenge 2020: International Declarations, Eindhoven University of Technology, Eindhoven, The Netherlands, 2020, http://dx.doi.org/10.4121/uuid:2bbf8f6a-fc50-48eb-aa9e-c4ea5ef7e8c5.

[58] F. Mannhardt, Bpi Challenge 2020: International Declarations, Eindhoven University of Technology, Eindhoven, The Netherlands, 2017, http://dx.doi.org/10.1007/978-3-319-59536-8_34.

[59] F. Mannhardt, D. Blinde, Analyzing the trajectories of patients with sepsis using process mining, in: Joint Proceedings of the Radar Tracks At the 18th International Working Conference on Business Process Modeling, Development and Support (BPMDS), and the 22nd International Working Conference on Evaluation and Modeling Methods for Systems Analysis and Development (EMMSAD), and the 8th International Workshop on Enterprise Modeling and Information Systems Architectures (EMISA) Co-Located with the 29th International Conference on Advanced Information Systems Engineering 2017 (CAiSE 2017), Essen, Germany, June (2017) 12-13, 1859 of CEUR Workshop Proceedings, 2017, pp. 72–80, CEUR-WS.org, https://ceur-ws.org/Vol-1859/bpmds-08-paper.pdf.

[60] C. Di Ciccio, M.L. Bernardi, M. Cimitile, F.M. Maggi, Generating event logs through the simulation of declare models, in: J. Barjis, R. Pergl, E. Babkin (Eds.), Enterprise and Organizational Modeling and Simulation - 11th International Workshop, EOMAS 2015, Held At CAiSE 2015, Stockholm, Sweden, June (2015) 8-9, Selected Papers, in: Lecture Notes in Business Information Processing, vol. 231, Springer, 2015, pp. 20–36, http://dx.doi.org/10.1007/978-3-319-24626-0_2.

[61] L. Tacke Genannt Unterberg, Simulated event log: Udon-ya, 2023, http://dx.doi.org/10.5281/zenodo.8037210, Dataset on Zenodo.

[62] H. Weytjens, J.D. Weerdt, Creating unbiased public benchmark datasets with data leakage prevention for predictive process monitoring, in: Business Process Management Workshops - BPM 2021 International Workshops, Rome, Italy, September (2021) 6-10, Revised Selected Papers, in: Lecture Notes in Business Information Processing, vol. 436, Springer, 2021, pp. 18–29, http://dx.doi.org/10.1007/978-3-030-94343-1_2.

[63] M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, J. Amer. Statist. Assoc. 32 (200) (1937) 675–701.

[64] P.B. Nemenyi, Distribution-Free Multiple Comparisons, Princeton University, 1963.

[65] Y. Yu, C.-H.H. Yang, J. Kolehmainen, P.G. Shivakumar, Y. Gu, S.R.R. Ren, Q. Luo, A. Gourav, I.-F. Chen, Y.-C. Liu, et al., Low-rank adaptation of large language model rescoring for parameter-efficient speech recognition, in: 2023 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU, IEEE, 2023, pp. 1–8.

[66] A. Rebmann, F.D. Schmidt, G. Glavaš, H. van der Aa, Process behavior corpus and benchmarking datasets, 2024, http://dx.doi.org/10.5281/zenodo.11276246.

[67] I. Loshchilov, F. Hutter, Fixing weight decay regularization in adam, 2017, CoRR abs/1711.05101, arXiv:1711.05101. http://arxiv.org/abs/1711.05101.

[68] E. Rama-Maneiro, J.C. Vidal, M. Lama, Deep learning for predictive business process monitoring: Review and benchmark, IEEE Trans. Serv. Comput. 16 (1) (2021) 739–756.

[69] W. Ni, G. Zhao, T. Liu, Q. Zeng, X. Xu, Predictive business process monitoring approach based on hierarchical transformer, Electronics 12 (6) (2023) 1273.

[70] A. Alibakhshi, E. Hassannayebi, Towards an enhanced next activity prediction using attention based neural networks, Clust. Comput. 28 (1) (2025) 54, http://dx.doi.org/10.1007/S10586-024-04830-8.

[71] M. Hinkka, T. Lehto, K. Heljanko, A. Jung, Classifying process instances using recurrent neural networks, in: F. Daniel, Q.Z. Sheng, H. Motahari (Eds.), Business Process Management Workshops - BPM 2018 International Workshops, Sydney, NSW, Australia, September (2018) 9-14, Revised Papers, in: Lecture Notes in Business Information Processing, vol. 342, Springer, 2018, pp. 313–324, http://dx.doi.org/10.1007/978-3-030-11641-5_25.

[72] M.A. Khan, H. Le, K. Do, T. Tran, A. Ghose, K.H. Dam, R. Sindhgatta, Memory-augmented neural networks for predictive process analytics, 2018, CoRR abs/1802.00938, arXiv:1802.00938.

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

[74] D. Impedovo, G. Pirlo, G. Semeraro, Next activity prediction: An application of shallow learning techniques against deep learning over the BPI challenge 2020, IEEE Access 11 (2023) 117947–117953, http://dx.doi.org/10.1109/ACCESS.2023.3325738.

[75] 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 (6) (2024) 4111–4124, http://dx.doi.org/10.1109/TSC.2024.3463487.

[76] A. Abane, A. Bekri, A. Battou, Fastrag: Retrieval augmented generation for semi-structured data, 2024, CoRR abs/2411.13773, arXiv:2411.13773.