SNAP: Semantic Stories for Next Activity Prediction

Alon Oved*, Segev Shlomov*, Sergey Zeltyn*, Nir Mashkif, Avi Yaeli

IBM Research - Israel
University of Haifa Campus, Mount Carmel, Haifa 3498825, Israel
alon.oved@ibm.com, segev.shlomov1@ibm.com, sergeyz@il.ibm.com, nirm@il.ibm.com, aviy@il.ibm.com

Abstract

Predicting the next activity in an ongoing process is one of the most common tasks in the domain of business process management (BPM). It allows businesses to optimize resource allocation, enhance operational efficiency, and aid both in risk mitigation and strategic decision-making. Existing state-of-the-art AI models for BPM do not fully capitalize on available semantic information within process event logs. As current advanced AI-BPM systems provide semantically richer textual data, the need for new adequate models grows. To address this gap, we develop SNAP—a novel system that utilizes LLMs by constructing narratives and semantic contextual stories for historical event logs, which are then used to generate precise and actionable predictions for the ongoing process. SNAP was evaluated on six benchmark datasets, where it demonstrated significant performance improvements over eleven SOTA models, particularly on datasets with high levels of semantic content. This work showcases the potential of integrating LLMs in BPM and outlines a clear path toward future deployment, emphasizing the relevance and innovation of our approach within the broader AI application landscape.

Introduction

Next activity prediction (NAP) of an ongoing process is a cornerstone in the BPM domain. It is embedded in all commercial process mining tools such as IBM Process Mining and Celonis, and enables organizations to enhance operational efficiency, mitigate risks, and support strategic decision making. NAP is mainly used in the following four applications: Next best action recommendation - advise the next best activity based on selected key performance indicators (Weinzierl et al. 2020); anomaly detection - identify a deviating process instance compared with the distribution of the next activity predictions (Nolle et al. 2022); resource allocation - proactively assign resources based on activity prediction (Park and Song 2019); and preemptive guidance - monitor the most plausible activities to flag inefficiencies, risks, and mistakes (Di Francescomarino and Ghidini 2022).

NAP has been widely explored using deep learning (DL) (Rama-Maneiro, Vidal, and Lama 2021), transformers (Vaswani et al. 2017), small language models (Devlin et al. 2019; He et al. 2021), and process mining techniques (Van der Aalst, Schonenberg, and Song 2011). These approaches are also applicable for a variety of predictive process monitoring tasks such as trace suffix prediction, process outcome prediction, and remaining time prediction (Neu, Lahann, and Fettke 2021).

Despite the advancements in AI applications for BPM, existing state-of-the-art (SOTA) models fail to fully utilize the semantic information embedded within process event logs. These models predominantly rely on sequences of activities while neglecting the contextual richness provided by numerical, categorical, and especially textual data. This oversight is particularly limiting in domains like robotic process automation (RPA) (Van der Aalst, Bichler, and Heinzl 2018), where conversational interfaces generate semantically rich data from user-bot interactions. In these systems, traces composed of semantic identifiers, selected skills, timestamps, utterances, and other resources can offer valuable insights if appropriately leveraged (Shlomov et al. 2024). The current trend of simplifying models by excluding these rich contexts ultimately restricts their predictive power and practical applicability (Chakraborti et al. 2020; Yaeli et al. 2022).

To address this gap, we introduce SNAP (Semantic Narratives for Next Activity Prediction), a novel system that capitalizes on the capabilities of Large Language Models (LLMs) to construct semantic contextual stories from historical event logs. Unlike traditional NAP models, SNAP integrates the richness of natural language to create coherent narratives that encapsulate the full scope of a business process. By transforming process data into these semantic stories, SNAP enhances prediction accuracy and provides actionable insights, particularly in scenarios with high levels of semantic content. SNAP's methodology involves several key steps: (1) Identifying and designing relevant features to be included in the narratives; (2) using an LLM to generate a story template that captures the essence of the business process; (3) transforming event logs into semantic stories that serve as input for the model; and (4) fine-tuning a pre-trained small language model (SLM) for next activity prediction as a classification task.

We conducted a comprehensive evaluation of SNAP on six benchmark datasets, comparing its performance against eleven state-of-the-art models. The results demonstrate that SNAP consistently outperforms these models, particularly

*Equal contribution.

in datasets rich in semantic content. In the context of conversational RPA, where the integration of textual and contextual data is crucial, SNAP achieved remarkable improvements in prediction accuracy. The contributions of this work are twofold. First, we introduce SNAP, a system that utilizes LLMs to transform business process data into semantic narratives for enhanced next activity prediction. Second, we demonstrate SNAP's superior performance on multiple BPM datasets, including a specialized dataset from the conversational RPA domain, showcasing SNAP's potential for real-world deployment. This research underscores the importance of integrating advanced AI techniques, such as LLMs, into BPM, while highlighting a clear path toward deploying these innovations in practical applications. The findings presented in this paper not only advance the state of the art in NAP, but also emphasize the broader relevance and innovation of our approach within the AI application landscape.

Related Work

Next activity prediction (NAP) and other BPM prediction tasks have a research history of several decades and numerous business applications. Initially, this research used classical machine learning (ML) methods; (Márquez-Chamorro, Resinas, and Ruiz-Cortés 2017) survey these techniques. Typically, ML methods, including SVM, Random Forests, and others, easily incorporate numeric and categorical attributes, but do not work well with sequential data and struggle with free-text and semantic information. Their performance is also typically inferior to DL methods.

DL models, such as LSTM, provided the best results for next activity prediction before the advent of transformers. (Neu, Lahann, and Fettke 2021) contains a detailed review of this domain and discusses other aspects of the prediction pipeline, such as feature engineering and encoding. These methods are well suited for relatively short sequential data, but still have trouble incorporating free-text features, semantics, and long-term dependencies. (Rama-Maneiro, Vidal, and Lama 2021) reviews and runs nine DL algorithms for NAP, including Gated Recurrent Unit, convolutional NN and LSTM, on 12 open-access BPM datasets.

Applying transformer models (Neu, Lahann, and Fettke 2023) to the PBPM domain and NAP task raises several technical challenges. Philipp et al. (2020) uses sequences of activities from event logs, where activity types are encoded to integers. An attempt to use BERT for NAP is shown in (Chen, Fang, and Fang 2022). Their method performs a pretraining phase with a masked activity model (MAM) task and adds a classification head for the fine-tune phase. Similarly to (Philipp et al. 2020), the input for the model is only the sequence of activities from the process trace. However, these models struggle to use semantic and textual content present in many advanced process logs, such as conversational RPA systems. In addition, they do not include many attributes in the input data due to state-space explosion.

Recent advances in goal-driven chatbots, RPAs (Chakraborti et al. 2020) and LLM-based agents (Guo et al. 2024) gave rise to a new class of business processes. (Rizk et al. 2020) describes an approach to design and orchestration of such systems. RPA systems extensively use chatbots and automation tools to interact with human process participants and automate business tasks. These applications include user utterances and bot responses in event logs, resulting in new types of activities. A conversation session with a chatbot constitutes a sub-process. The combination of a user utterance and a chatbot response can be considered a basic activity implying a new level of semantic richness in the process. (Zeltyn et al. 2022) introduced a conversational RPA dataset based on an actual use-case in the Human Resources domain.

SNAP System Framework

Problem description The building blocks of BPM include activity, case ID, traces, and others. These elements form the basic structure of a BPM system and are essential for defining, modeling, and executing business processes. Activities are units of work executed within a process. Activities can be sequential or parallel and can be defined using various attributes such as roles, resources, and dependencies. These attributes play a significant role in the process and may contain free text values. The case ID is a unique identifier assigned to each instance of a process. It allows organizations to track and manage individual cases and provides a way to aggregate data and analyze process performance. A trace consists of a finite sequence of events executed in a case, each defined by an activity, timestamp, and several attributes. An event log is the set of all traces executed in a process.

Next activity prediction - An event with L different attributes is defined as: $e = (a, t, (d_1, v_1), ..., (d_L, v_L))$, where a is the activity, t is its timestamp, d_i is an attribute name, and v_i is the corresponding attribute value. Given a trace prefix such that $p_k = \langle e_1, e_2, ..., e_k \rangle$, we aim to predict the next event's activity, a_{k+1}, from the trace prefix (i.e., $f(p_k) = a_{k+1}$).

SNAP

The main idea of the SNAP model is based on the concept of semantic stories. Through narratives and special structure, stories include semantics and meaning, making them universal elements of the human experience. In the context of processes, stories can help describe and structure a process and add meaning to the process data. Thus, to utilize all the available information in the process event log, we construct a story that captures the semantic information of each prefix trace. It provides a unified textual representation of the traces that allow us to fine-tune SLMs and achieve state-of-the-art results. The architecture of SNAP is presented in Figure 1.

List of attributes - The original event logs consist of L different attributes. In many cases, only a few features should be part of the semantic story. There are two main reasons for that assertion. First, an SLM is limited in its input size (e.g., the 512 token limit in BERT), and second, we argue and empirically test that not all attributes are statistically significant for the classification task. We suggest using an XGBoost model for feature selection and set a threshold on the attributes' importance. Temporal features are computed from trace timestamps. For example, the time from the case

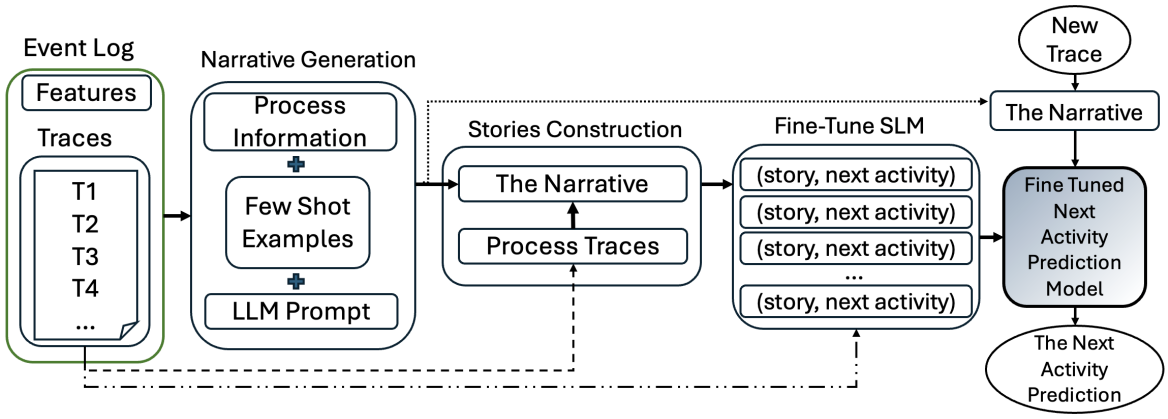

Figure 1: SNAP architecture for story-based classification in BPM. Starting with an event log as input, SNAP generates a single-story narrative by prompting an LLM with few-shot examples and process information. It then pours traces into this narrative to create semantic stories, which, along with their next actions, are used to fine-tune a small language model. When a new trace is provided, SNAP first transforms it into a story using the generated narrative and then employs the fine-tuned model to predict the next activity.

start and the time from the previous activity is computed and added to the features list as a new temporal feature.

Story narrative construction - To transform each prefix trace into a story, we first construct a story narrative. To do so, we utilize an LLM (e.g., Llama 3 (Touvron et al. 2023)), as these models excel at such tasks. Given the list of features, SNAP uses an LLM prompt to generate a narrative story. We recommend using at least a one-shot example of an input-output pair in the prompt, which can easily be modified to a specific required domain. Table 2 illustrates an example of the one-shot prompt we used.

Since the story narrative construction is performed by an LLM, we advise reviewing its result and verifying its coherence. In some instances, we might prefer a shorter narrative or the removal of unwanted information. In such cases, we can then manually edit it. Based on the output narrative generated by the LLM, we implement a deterministic function $N : P \rightarrow S$, where P is the set of all process traces and S is the set of all natural language stories.

Semantic story transformation - Utilizing the narrative function N, we transform every prefix trace P_k in the event log E into a story s_k. It is important to note that this step is deterministic. Generative LLM is used only once in the previous step. An example of a prefix trace of a loan application contains the following features (or values): Activity (Register Application), Turn (6), Time from case start (12 days), Support line (Customer Support), Latest impact (Moderate), Owner country (USA), Involved ST function div (Finance Division), Involved ST (John Doe) and Product (Personal loan). The generated story result is depicted in Table 1.

We note that the number of tokens in the story may exceed an SLM's limit for tokens, especially when fine-tuning small models such as BERT. That is amplified when the largest case length of several datasets surpasses 150 activities. To mitigate this, SNAP imposes a maximum trace length limit for the number of previous activities we include in the story.

Fine-tuning SLM - The transformation of the event log into stories produces a dataset containing sample-label pairs $\{(s_k, a_{k+1}) : \forall p_k \in E\}$. Each sample represents a story s_k of a prefix trace, and the label a_{k+1} represents the corresponding next activity ground-truth. With this story-based dataset in hand, we fine-tune small language models (BERT and DeBERTa) with their compatible tokenizers on the next-activity classification task.

The requested loan amount was $20,000, and it was requested by the customer. The activity "register Application" took place on turn 6, which occurred 12 days after the case started. The support line responsible for handling this application was the Customer Support team. The latest impact of the loan application was assessed as "Moderate". The loan application was under the ownership of the United States, indicating that the applicant resided in the USA. The finance division was involved in processing this application, with John Doe taking responsibility for it. The specific product being applied for was a Personal Loan. The sequence of recent activities leading up to this event includes the following steps: preliminary checks, document verification, credit assessment, and application review.

Table 1: A semantic story for a loan application example. The prompt in Table 2 was used to generate the narrative and thereafter the trace was transformed into this semantic story.

Experimental Results

Datasets and benchmarks Table 3 describes six publicly available datasets that are used for the validation of our prediction models. All datasets, except MIP, are real-world event logs that can be downloaded from the *4TU Center for Research Data (*data.4tu.nl/info/en/*)*. The MIP synthetic

You are given a list of features that were extracted from a process event log. Your task is to create a semantic story narrative that will best represent traces in the event log. Here is an example:

Input:

Features: activity, request start date, time to current activity, amount, resource, previous activities

Domain: bank loans

Feature value examples:

- activity: under review
- request start date: Aug 20, 2023
- time to current activity: 12 days
- amount: $20000
- resource: 10629

Output:

A customer applied for a personal loan with an amount of [*amount_placeholder*] on [*request_start_date_placeholder*]. Since the start of the request, [*time_to_current_activity*] has elapsed. The loan application is currently [*activity_placeholder*]. This process is being handled by the resource identified as [resource_placeholder]. Prior to this stage, the application has undergone various activities, including [*history_of_past_activities_placeholder*]. Each activity in this journey contributes to a comprehensive evaluation aimed at determining the applicant's eligibility and the loan's viability.

Generate your response for the following input

Input:

Features: {list of features}

Domain: {event-log domain}

Feature value examples: {list of features values example}

Table 2: The LLM prompt used to create a story narrative with a one-shot example. The output is a narrative with its corresponding parameters placeholders (marked as [*_*placeholder*]).

dataset was presented and published in (Zeltyn et al. 2022). The benchmark datasets cover a wide range of domains. The two BPI13 logs describe Volvo IT procedures for closed problems and incidents. The Env Permit is related to an environmental permit application process, while the Sepsis event logs represent the pathway of patients with sepsis through a hospital. The NASA dataset was obtained through the code instrumentation tools and describes events and exceptions in a software system. Finally, the MIP (Management Incentive Program) dataset has been simulated to resemble a real-world HR use-case in the conversational RPA domain.

The selection of datasets in Table 3 was strongly affected by (Rama-Maneiro, Vidal, and Lama 2021). We validate the novel SNAP algorithm on several event logs analyzed in this paper. Specifically, we selected datasets that contain activity names and attributes with semantic meaning.

Semantic stories on benchmark datasets

According to the SNAP description above, we start with the feature selection step, applying XGBoost for the NAP task using all available features. We then select up to six features with the importance score above a set threshold. XGBoost cannot incorporate free-text attributes with practically limitless possible values (e.g., user utterances in MIP). We therefore added them to the selected features list.

Dataset	Num. cases	Num. events	Num. activities	Avg. case length
BPI13cp	1487	6660	7	4.48
BPI13in	7554	65533	13	8.68
Env Permit	1434	8577	27	5.98
Sepsis	1049	15214	16	14.48
Nasa	2566	73638	94	28.70
MIP	1000	49604	36	49.60

Table 3: Description of benchmark datasets

After the feature selection stage, the Llama-2 language model (Touvron et al. 2023) was applied to generate the story template with the prompt described in Table 2.

Dataset Example
Features in the event log: session number, role, user id, timestamp, activity, turn, user utterance, chatbot response, intent, intent confidence, entity, entity confidence, score, expecting response **Selected features:** role, turn number, session number, user utterance, chatbot response, sequence of skills **Semantic story example:** During the third session, a *team leader* initiated a request, marking the fourth *turn* of the conversation. The *request* was to "view project assessment report", reflecting the user's intent to scrutinize project-related performances. The chatbot, in *response*, presented the "project assessment report", aligning its reply with the user's demand. This interaction is part of a broader *conversational flow* that began with a welcoming message, proceeded to report yearly assessments, engaged in disambiguation to clarify queries, and culminated in reporting project assessments. This sequence showcases the chatbot's adeptness in navigating through a sequence of skills, tailored to address the evolving needs of its users in the context of HR candidate promotion. **Trace label:** *Report learning activities*

Table 4: MIP dataset: Illustration to story design pipeline

Table 4 explains the pipeline of the story template design using the MIP dataset. The user, a manager of a software engineering team, engages with the system to view different performance reports and, subsequently, to increase a salary of employees with the best performance. We select features from the log and generate the story template using LLM. Event log data is then deterministically transformed using the story template to acquire the processed dataset that consists of pairs of textual semantic stories and their labels (actual next activities). We then train the fine-tune phase of the pre-trained SLM for a classification task.

Fine-tuning setup and implementation

In our fine-tuning SNAP experiments with bi-directional language models, we run two prediction algorithms based

on BERT and DeBERTa, respectively. The experiment setup is consistent with the benchmark paper (Rama-Maneiro, Vidal, and Lama 2021). We used 5-fold cross-validation with a 64-16-20 train-validation-test split.

For each algorithm and dataset, we computed the average accuracy and F1 score that were weighted by the number of cases in each class. Dropout rate was equal to 0.5. Learning rate, batch size and maximum window backward trace length were optimized on the validation sets. Ultimately, for 4 datasets out of 6, the optimal hyper-parameters were batches of size 4, learning rate of 10^{-5}, and a window size equal to 10 past activities. A window of 15 activities has been optimal for Sepsis, and a batch of size 8 was used for MIP. A maximum number of 15 epochs were run for each experiment. The training stopped beforehand if validation accuracy did not increase for three consecutive epochs. The model with the best validation accuracy was saved and validated on the testing set. The algorithms with bi-directional LM were run on x86 compute nodes with Nvidia V100 GPU and an 80GB memory requirement. Experiments with GPT-3 fine-tuning were performed using the OpenAI API.

Results

Comparing SNAP and the state of the art. In our main validation experiment, we fine-tuned several well-known SLMs and compared their performance with the benchmarks. First, we considered two bi-directional SLMs, BERT - *bert-base-cased* and DeBERTa - *microsoft/deberta-base*, and named their implementations SNAP-B and SNAP-D, respectively. Then we fine-tuned OpenAI GPT-3 model and named it SNAP-G. GPT-3 model is considered much more powerful than BERT and its modifications, so we tested the performance differences between SNAP-G and the other two models. Tables 5 and 6 present the results of the experiment.

The state-of-the-art performance is taken from (Rama-Maneiro, Vidal, and Lama 2021, 2023), where TACO is the original algorithm in (Rama-Maneiro, Vidal, and Lama 2023). The latter paper does not contain results for F1-score and does not analyze the NASA and Sepsis datasets. The MIP benchmark is taken from (Zeltyn et al. 2022).

We performed the non-parametric Wilcoxon signed-rank test to check the statistical significance of the results (Dror et al. 2018, 2020). We separately tested each SNAP model against the SOTA. In terms of F1-score (Table 6), SNAP significantly outperforms the SOTA on all datasets. Our empirical analysis reveals that SNAP-G accuracy yields superior outcomes over most of the datasets (Table 5). SNAP-B is slightly better than SNAP-G for NASA and TACO is slightly better for BPI13in. In Table 6, we can see that SNAP-G's F1-score outperforms other methods for all datasets, except NASA.

As SNAP-G is a super-large model and the stories contain all the process information, it seems plausible that its results are close to the maximal achievable accuracy. Comparing SNAP-G to SNAP-B and SNAP-D, we observe that even the smaller SNAP models demonstrate solid performance. Comparing SNAP-B and SNAP-D shows that in instances where SNAP-D demonstrates higher performance than SNAP-B, the discernible advantage is of modest proportion. Conversely, in scenarios where SNAP-B outperforms SNAP-D, such as in the case of the NASA and Sepsis datasets, the observed performance difference is substantial.

	BPI13 cp	BPI13 in	Env	Sepsis	Nasa	MIP
Camargo	0.547	0.667	0.858			
Evermann	0.588	0.668	0.762	0.400	0.204	
Hinnka	0.635	0.747	0.844	0.635	0.885	
Khan	0.436	0.519	0.836	0.210	0.127	
Mauro	0.249	0.367	0.536	0.615	0.210	
Pasquadibisceglie	0.475	0.460	0.867	0.562	0.883	
Tax	0.640	0.701	0.857	0.642	0.894	
Theis	0.595	0.594	0.863	0.557	0.890	
Venugopal	0.484	0.496	0.696			
TACO	0.675	**0.777**	0.877			
Zeltyn						0.390
SNAP-B	0.679*	0.758	0.880	0.649*	**0.898***	0.424*
SNAP-D	0.679*	0.764	0.873	0.567	0.846	0.437*
SNAP-G	**0.696***	0.774	**0.890***	**0.655***	0.895	**0.459***

Table 5: Accuracy comparison between the SNAP-B, SNAP-D, SNAP-G, and SOTA benchmark algorithms. The best, second-best, and third-best approaches are highlighted in cyan, orange, and yellow, respectively. *The SNAP models significantly outperform the SOTA with $p_{value} < 0.05$ (Wilcoxon signed-rank test).

	BPI13 cp	BPI13 in	Env	Sepsis	Nasa	MIP
Camargo	0.523	0.614	0.851			
Evermann	0.505	0.585	0.739	0.303	0.123	
Hinnka	0.571	0.730	0.825	0.612	0.877	
Khan	0.369	0.434	0.822	0.176	0.106	
Mauro	0.100	0.362	0.524	0.602	0.169	
Pasquadibisceglie	0.427	0.374	0.849	0.536	0.874	
Tax	0.590	0.684	0.842	0.633	0.886	
Theis	0.534	0.547	0.845	0.526	0.880	
Zeltyn						0.330
SNAP-B	0.670*	0.754*	0.866*	0.638*	**0.890***	0.421*
SNAP-D	0.664*	0.757*	0.860*	0.540	0.832	0.428*
SNAP-G	**0.678***	**0.767***	**0.876***	**0.641***	0.887	**0.447***

Table 6: Weighted F1-score comparison between the SNAP-B, SNAP-D, SNAP-G, and SOTA benchmark algorithms. The best, second-best, and third-best approaches are highlighted in cyan, orange, and yellow, respectively. *The SNAP models significantly outperform the SOTA with $p_{value} < 0.05$ (Wilcoxon signed-rank test).

The relatively low accuracy numbers for the MIP dataset are due to the large number of activities and the significant variance in the sequences it contains. Yet, SNAP significantly outperforms the benchmark models due to a large amount of semantic information in MIP. Furthermore, we see that the average enhancement margin of SNAP models is more substantial for the F1-score compared to accuracy. Therefore, SNAP seems to be more balanced between recall and precision than the state-of-the-art algorithms.

Does semantic story structure matter? Design of coherent and grammatically correct semantic stories from business process logs constitutes a key step in the SNAP algorithm. It is natural to ask if this step is necessary. One can

simply concatenate feature values, including the historical sequence of activities, into a text string and use it a "basic story" input to the SNAP algorithm.

Dataset	Semantic story		List of values		Numbered activities	
	acc.	F1	acc.	F1	acc.	F1
BPI13cp	0.679	0.670	0.616	0.614	0.670	0.661
BPI13in	0.758	0.754	0.733	0.731	0.758	0.753
Env Perm.	0.880	0.866	0.859	0.845	0.877	0.865
Sepsis	0.649	0.638	0.635	0.630	0.648	0.640
Nasa	0.896	0.890	0.880	0.877	0.896	0.889
MIP	0.424	0.421	0.426	0.421	0.401	0.395

Table 7: SNAP-B comparison: Semantic stories versus lists of feature values and stories with numbered activities

Table 7 includes comparison of the performance metrics for the two approaches using the same 5-fold fine-tuning setup. We observe that the semantic representation significantly improves the accuracy and F1-score of our approach for all datasets except MIP, where they are comparable.

Does semantic information in activity names matter? SNAP, gives meaningful names to the activities and performs some preprocessing work to replace shortened or unclear activity names in the logs. We tested the model's performance when the activity names were replaced by integers in the stories, thus removing their semantic content.

Table 7 includes SNAP-B comparison for the two cases. We observe a significant performance decline for MIP, a moderate decline for BPI13cp and Env Permit and non-significant decline for other datasets. We conclude that the importance of semantically meaningful activity names strongly depends on the dataset. For example, in the case of MIP dataset, semantic meaning of activity names is more important that grammatically correct and coherent story. In other cases, SNAP can be applicable with satisfactory results even if meaningful activity names are unavailable.

Do user utterances in conversational RPA system matter? In contrast to classical business processes, the logs of conversational RPA systems contain rich textual information of the user utterances and bot responses. We checked if this information is vital for prediction quality and ran SNAP algorithm with BERT and DeBERTa models excluding user utterances on MIP dataset. The best results were achieved via DeBERTa with accuracy of 0.373 and weighted F1-score of 0.358, significantly inferior to 0.437 and 0.428, respectively, in Tables 5 and 6. These experiments, jointly with Table 7, indicate that various types of semantic information within the logs pose meaningful value for our predictions. They also demonstrate that SNAP can be especially relevant for logs with rich textual and conversational data.

Discussion and Deployment

SNAP is a novel next activity prediction approach based on constructing semantic stories from event logs. The core concept of semantic stories presents a highly efficient methodology for incorporating activities and their valuable event-log attributes into a cohesive narrative. This approach is particularly useful when the categorical feature space is huge, such as user utterances and other free-text attributes. The usage of small BERT-like models with the stories results in a semantic bi-directorial encoding of information of the trace. Through comprehensive evaluation, we demonstrate that SNAP outperforms existing state-of-the-art models.

The SNAP model's performance is limited by the level of semantic information in event logs, particularly when attributes are mainly numeric. In such cases, traditional ML techniques may be more effective, as SNAP's effectiveness increases with richer semantic data, especially in conversational RPA systems and digital assistants. While the SNAP model shows statistically significant improvements over the SOTA, the average enhancement of around 7% may seem modest, especially given the challenges in a mature field like BPM. However, SNAP's advantage is expected to grow dramatically with the rise of AI applications and semantically rich datasets.

Deployment SNAP is a Python-based framework that fine-tunes SLMs using a single API call to an LLM, providing an efficient approach to AI-driven process management. The system is designed with modularity, allowing it to easily adapt to various deployment environments. By leveraging Kubernetes-managed models (K-models), SNAP hosts lightweight, open-source SLMs that are easy to maintain and update. This architecture enhances the system's flexibility and scalability across cloud platforms such as AWS, Azure, and IBM Cloud. Docker further supports consistent deployment and seamless migration across development, testing, and production stages, ensuring SNAP remains reliable under various conditions. The use of K-models also enables continuous training, allowing SNAP to dynamically adapt to new data and evolving business needs, keeping the system responsive and up-to-date.

SNAP is being prepared for its initial integration into the process mining ecosystem, with plans to test it within IBM Process Mining. This testing phase is critical for assessing the system's performance and compatibility. The modular design of SNAP ensures that it can be easily deployed across a range of process mining platforms, offering significant potential for widespread adoption. Furthermore, SNAP has the potential to be deployed in conversational RPA tools like watsonx Assistant for next-activity prediction, expanding its applicability in AI-driven process management. SNAP is also highly relevant to the integration between process mining and conversational bots like IBM watsonx Orchestrate, helping to create synergy between them. This capability enhances process management in the new area of AI and facilitates conversation agents via process mining techniques, making SNAP a versatile tool in the broader AI and RPA landscape.

References

Chakraborti, T.; Isahagian, V.; Khalaf, R.; Khazaeni, Y.; Muthusamy, V.; Rizk, Y.; and Unuvar, M. 2020. From

robotic process automation to intelligent process automation. In *BPM 2020 Blockchain and RPA Forum*, 215–228. Springer, Cham.

Chen, H.; Fang, X.; and Fang, H. 2022. Multi-task prediction method of business process based on BERT and Transfer Learning. *Knowledge-Based Systems*, 254: 109603.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL-HLT*, 1: 4171–4186.

Di Francescomarino, C.; and Ghidini, C. 2022. Predictive process monitoring. *Process Mining Handbook. LNBIP*, 448: 320–346.

Dror, R.; Baumer, G.; Shlomov, S.; and Reichart, R. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *Proceedings of the 56th annual meeting of the association for computational linguistics (volume 1: Long papers)*, 1383–1392.

Dror, R.; Peled-Cohen, L.; Shlomov, S.; and Reichart, R. 2020. *Statistical significance testing for natural language processing*. Springer.

Guo, T.; Chen, X.; Wang, Y.; Chang, R.; Pei, S.; Chawla, N. V.; Wiest, O.; and Zhang, X. 2024. Large language model based multi-agents: A survey of progress and challenges. *arXiv preprint arXiv:2402.01680*.

He, P.; Liu, X.; Gao, J.; and Chen, W. 2021. Deberta: Decoding-enhanced bert with disentangled attention. *International Conference on Learning Representations (ICLR)*.

Márquez-Chamorro, A. E.; Resinas, M.; and Ruiz-Cortés, A. 2017. Predictive monitoring of business processes: a survey. *IEEE Transactions on Services Computing*, 11(6): 962–977.

Neu, D. A.; Lahann, J.; and Fettke, P. 2021. A systematic literature review on state-of-the-art deep learning methods for process prediction. *Artificial Intelligence Review*, 1–27.

Neu, D. A.; Lahann, J.; and Fettke, P. 2023. Predictive Business Process Monitoring Approach Based on Hierarchical Transformer. *Electronics*, 12(6): 1273.

Nolle, T.; Luettgen, S.; Seeliger, A.; and Mühlhäuser, M. 2022. Binet: Multi-perspective business process anomaly classification. *Information Systems*, 103: 101458.

Park, G.; and Song, M. 2019. Prediction-based resource allocation using LSTM and minimum cost and maximum flow algorithm. In *2019 international conference on process mining (ICPM)*, 121–128. IEEE.

Philipp, P.; Jacob, R.; Robert, S.; and Beyerer, J. 2020. Predictive analysis of business processes using neural networks with attention mechanism. In *2020 International conference on artificial intelligence in information and communication (ICAIIC)*, 225–230. IEEE.

Rama-Maneiro, E.; Vidal, J.; and Lama, M. 2021. Deep learning for predictive business process monitoring: Review and benchmark. *IEEE Transactions on Services Computing*.

Rama-Maneiro, E.; Vidal, J.; and Lama, M. 2023. Embedding graph convolutional networks in recurrent neural networks for predictive monitoring. *IEEE Transactions on Knowledge and Data Engineering*.

Rizk, Y.; Isahagian, V.; Boag, S.; Khazaeni, Y.; Unuvar, M.; Muthusamy, V.; and Khalaf, R. 2020. A Conversational Digital Assistant for Intelligent Process Automation. In *BPM 2020 Blockchain and RPA Forum*, 85–100. Springer, Cham.

Shlomov, S.; Yaeli, A.; Marreed, S.; Schwartz, S.; Eder, N.; Akrabi, O.; and Zeltyn, S. 2024. IDA: Breaking Barriers in No-code UI Automation Through Large Language Models and Human-Centric Design. *arXiv preprint arXiv:2407.15673*.

Touvron, H.; Martin, L.; Stone, K.; Albert, P.; Almahairi, A.; Babaei, Y.; Bashlykov, N.; et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Van der Aalst, W. M.; Bichler, M.; and Heinzl, A. 2018. Robotic process automation. *Business & information systems engineering*, 60(4): 269–272.

Van der Aalst, W. M.; Schonenberg, M. H.; and Song, M. 2011. Time prediction based on process mining. *Information systems*, 36(2): 450–475.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Weinzierl, S.; Stierle, M.; Zilker, S.; and Matzner, M. 2020. A next click recommender system for web-based service analytics with context-aware LSTMs. *53rd Hawaii International Conference on System Sciences (HICSS)*, 1–10.

Yaeli, A.; Shlomov, S.; Oved, A.; Zeltyn, S.; and Mashkif, N. 2022. Recommending Next Best Skill in Conversational Robotic Process Automation. In *International Conference on Business Process Management*, 215–230. Springer.

Zeltyn, S.; Shlomov, S.; Yaeli, A.; and Oved, A. 2022. Prescriptive Process Monitoring in Intelligent Process Automation with Chatbot Orchestration. *IJCAI 2022 International Workshop on Process Management in the AI era*, 49–60.