Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Model-Semantics-Aware Predictive Process Monitoring

Dinh-Lam Pham
Kyonggi University
Jinyeong Lee
Hanyang University
Tsagaantsooj Batzaya
Kyonggi University
Hyun Ahn
Hanshin University
Minjae Park
Daelim University College
Ilkyeun Ra
University of Colorado Denver
Kwanghoon Kim
kwang@kgu.ac.kr

Kyonggi University

Research Article

Keywords: Predictive process monitoring (PPM), Next process-activity prediction, Transformer neural network architecture, LSTM neural network architecture, Process event log datasets, Process mining, Model-semantics-aware process trace preprocessing, Business process management (BPM)

Posted Date: September 30th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10884433/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Model-Semantics-Aware Predictive Process Monitoring

Dinh-Lam Pham[1†], Jinhyeong Lee[2†], Tsagaantsooj Batzaya[3†], Thanh-Hai Nguyen[4†], Hyun Ahn[5*†], Minjae Park[6*†], Ilkyeun Ra[7*†], Kwanghoon Pio Kim[1,2,8*†]

[1]Contents Convergence Software Research Institute, Kyonggi University, Suwon, 16227, Gyeonggi, South Korea.
[2]Department of Artificial Intelligence, Graduate School of Hanyang University, Seoul, 04763, Seoul, South Korea.
[3]Faculty of Information Technology, Thai Nguyen University of Information and Communication Technology, Thai Nguyen, 250000, Thai Nguyen, Vietnam.
[4*]School of Computing and Artificial Intelligence, Hanshin University, Oasan, 18101, Gyeonggi, South Korea.
[5*]Department of Software, Daelim University, Anyang, 13916, Gyeonggi, South Korea.
[6]Department of Semiconductor System Engineering, Inha University, Incheon, 22212, Incheon, South Korea.
[7*]Department of Computer Science & Engineering, University of Colorado Denver, Denver, 80204, Colorado, USA.
[8*]School of AI Computer Science & Engineering, Kyonggi University, Suwon, 16227, Gyeonggi, South Korea.

*Corresponding author(s). E-mail(s): hahn@hs.ac.kr; mjpark@daelim.ac.kr; ilkyeun.ra@ucdenver.edu; kwang@kgu.ac.kr;
Contributing authors: phamdinhlam@kgu.ac.kr; jaims453@hanyang.ac.kr; tsbaachka95@kgu.ac.kr; nthai@ictu.edu.vn;
†These authors contributed equally to this work.

Abstract

Predictive Process Monitoring (PPM) increasingly employs deep learning to predict future process behavior from historical event logs. However, existing

approaches remain largely *log-centric*, learning from system-enacted traces that represent individual executions rather than the behavioral semantics defined by the underlying process model. This semantic gap may cause predictive models to learn execution-specific artifacts instead of process-relevant dependencies. This paper proposes *Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM)*, a new paradigm that explicitly integrates model-defined process semantics into predictive learning. MSA-PPM introduces a *Process Semantics Dictionary (PSD)* for representing parallel, exclusive-choice, loop, synchronization, temporal, resource, organizational, and constraint semantics. Guided by the PSD, conventional event traces are transformed into *model-semantics-aware process traces* through semantic preprocessing prior to predictive learning, thereby separating semantic interpretation from predictive models and architectures.
As an operational validation, we implement the parallel-semantics module of MSA-PPM. An exact Structured Information Control Net (SICN)-oriented process model is discovered using the $\boldsymbol{\rho}$-Algorithm, and its parallel semantics are used to identify and eliminate directed phantom edges caused by arbitrary execution orders among parallel sibling activities. The semantically revised traces are evaluated for next-activity prediction using LSTM and Transformer models. Experimental results show absolute accuracy improvements of 5.63 and 4.90 percentage points, respectively, together with improvements in Precision, Recall, and F1-score. These results demonstrate that explicitly incorporating model-defined process semantics can improve predictive performance while providing an architecture-independent semantic foundation for PPM.

Keywords: Predictive process monitoring (PPM); Next process-activity prediction; Transformer neural network architecture; LSTM neural network architecture; Process event log datasets; Process mining; Model-semantics-aware process trace preprocessing; Business process management (BPM)

1 Introduction

Predictive Process Monitoring (PPM) has emerged as an important branch of Business Process Management (BPM), combining process mining and machine/deep learning to predict future process behavior from historical event logs [1, 2]. Recent advances in Long Short-Term Memory (LSTM) networks, Transformers, Graph Neural Networks (GNNs), attention mechanisms, and hybrid architectures have substantially improved predictive performance for tasks such as next-activity, remaining-time, outcome, and risk prediction [3–5]. These capabilities enable proactive process management by supporting timely intervention, resource planning, and operational decision making.

Despite these advances, existing PPM approaches remain predominantly *log-centric*: predictive models learn directly from system-enacted process traces under the implicit assumption that observed event sequences adequately represent the semantics of the underlying business process. However, event logs capture individual executions, whereas process models define the behavioral semantics governing all valid executions. Consequently, observed activity orders and execution contexts may contain execution-specific artifacts that do not represent genuine process dependencies.

This discrepancy reveals a *semantic gap* between *system-enacted process traces* and *model-defined process semantics*. Process semantics encompass not only sequential dependencies but also parallelism, exclusive choice, loops, synchronization, temporal relationships, resources, organizational structures, and business constraints. Such semantics are generally only partially observable from individual event traces. Learning directly from these traces may therefore cause predictive models to capture incidental execution orders rather than the structural dependencies governing process behavior.

To address this limitation, this paper proposes *Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM)*, a paradigm that explicitly incorporates model-defined semantics into predictive learning. MSA-PPM introduces a *Process Semantics Dictionary (PSD)* as a machine-processable representation of process semantics. The PSD is designed to accommodate parallel, exclusive-choice, loop, synchronization, temporal, resource, organizational, and constraint semantics. Guided by this semantic knowledge, conventional event traces are transformed into *model-semantics-aware process traces* before predictive learning. This separation of semantic preprocessing from predictive modeling allows MSA-PPM to operate independently of particular predictive tasks and neural network architectures.

As an operational realization of MSA-PPM, this paper focuses on *parallel semantics*. An exact Structured Information Control Net (SICN)-oriented process model is discovered using the ρ-Algorithm, and its parallel structures are represented in the PSD. These semantics guide a *Phantom-Edge-Aware Semantic Preprocessing* method that identifies and eliminates directed phantom edges caused by arbitrary execution orders among parallel sibling activities while preserving genuine process dependencies. The resulting semantically revised traces are evaluated for next-activity prediction using representative LSTM- and Transformer-based PPM models.

The main contributions of this paper are summarized as follows:

- We propose *MSA-PPM*, which integrates model-defined process semantics into predictive learning through semantic preprocessing of system-enacted event traces.
- We introduce the *Process Semantics Dictionary (PSD)*, a modular representation for organizing model-defined semantics as reusable knowledge for predictive process monitoring.
- We develop a *Phantom-Edge-Aware Semantic Preprocessing* method as the parallel-semantics realization of MSA-PPM, transforming conventional event traces into model-semantics-aware traces without modifying the underlying predictive architectures.
- We validate the approach through next-activity prediction experiments with LSTM and Transformer models, demonstrating consistent performance improvements from incorporating model-defined parallel semantics.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the MSA-PPM paradigm and the Process Semantics Dictionary. Section 4 describes the parallel-semantics realization and its experimental evaluation for next-activity prediction. Finally, Section 5 concludes the paper and outlines future research directions.

2 Related Work and Research Gap

This section reviews related work from three perspectives: deep-learning-based predictive process monitoring, process semantics and model-aware prediction, and the research gap motivating Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM).

2.1 Deep-Learning-Based Predictive Process Monitoring

Predictive Process Monitoring (PPM) predicts the future behavior of ongoing process instances from historical event logs [5, 6]. Representative tasks include next-activity prediction [7], remaining-time prediction [3], outcome prediction [8, 9], anomaly prediction [10], and future-sequence prediction [2, 11]. Most approaches construct training samples by decomposing event traces into prefixes and encoding activity labels, timestamps, resources, and other attributes as numerical features.

Deep learning has substantially advanced this field. LSTM-based models have been widely employed to learn temporal dependencies from variable-length process traces [2, 3, 12], while Transformer-based models exploit self-attention to capture longer-range dependencies [1, 4, 13]. Recent studies have further investigated attention mechanisms, graph neural networks, explainable models, and hybrid architectures [14–17].

Despite these advances, most PPM approaches remain *log-centric*: predictive inputs are constructed directly from observed event sequences. Improvements therefore primarily concern encoding methods and learning architectures, while the observed ordering of events is generally assumed to represent valid process dependencies. This assumption can be problematic because an event trace represents a particular execution rather than the complete behavioral semantics of the underlying process model.

2.2 Process Semantics and Model-Aware Predictive Learning

Business process models explicitly represent behavioral relationships that cannot always be inferred from individual traces, including sequential, parallel, exclusive-choice, loop, and synchronization structures, together with temporal, resource, organizational, and business constraints. Process mining provides techniques for discovering such structures from event logs through algorithms such as the α-Algorithm [18], Heuristic Miner [19], Inductive Miner [20], and ρ-Algorithm [21, 22]. In particular, the ρ-Algorithm discovers Structured Information Control Net (SICN)-oriented process models composed of sequential, disjunctive, conjunctive, and iterative process patterns [22, 23].

Previous studies have also attempted to incorporate process and semantic knowledge into predictive learning. Representative approaches enrich event prefixes with contextual attributes, exploit process-model states or conformance information, employ graph-structured representations, or incorporate semantic and domain knowledge [14, 15, 24–26]. Such studies demonstrate that information beyond raw activity sequences can improve predictive performance and interpretability.

However, these approaches typically augment predictive inputs with additional features or incorporate model knowledge into a particular predictive architecture. Consequently, the original sequential representation of an event trace generally remains unchanged. Semantic enrichment therefore does not necessarily eliminate relationships that are present in an observed trace but inconsistent with the underlying process semantics. For example, parallel sibling activities may appear in different execution orders, causing a sequential representation to encode arbitrary interleavings as directed dependencies even though no such causal relationships exist in the process model.

2.3 Research Gap and Positioning of MSA-PPM

The literature therefore reveals a gap between *system-enacted process traces* and *model-defined process semantics*. Process mining can discover structural and behavioral knowledge, whereas PPM predominantly learns from observed execution sequences. Existing semantic or model-aware approaches provide valuable contextual information but do not generally establish an architecture-independent semantic layer for systematically transforming predictive traces according to model-defined semantics. MSA-PPM addresses this gap by treating process semantics as explicit knowledge applied *before* predictive learning. It introduces a *Process Semantics Dictionary (PSD)* as a modular representation of model-defined semantics and uses this knowledge to transform conventional event traces into *model-semantics-aware process traces*. The intended PSD encompasses parallel, exclusive-choice, loop, synchronization, temporal, resource, organizational, and constraint semantics, while remaining independent of a particular predictive task or learning architecture.

This study operationalizes and evaluates the *parallel-semantics* component of this broader paradigm. An SICN-oriented process model discovered by the ρ-Algorithm provides the parallel sibling relationships stored in the PSD. These relationships guide *Phantom-Edge-Aware Semantic Preprocessing*, which eliminates directed phantom edges caused by arbitrary ordering among parallel sibling activities while preserving model-defined dependencies [22, 27]. The resulting traces are evaluated for next-activity prediction using LSTM and Transformer models without modifying their internal architectures. Thus, the contribution evaluated in this paper is specifically the parallel-semantics realization of MSA-PPM, while the remaining PSD semantic categories define directions for extending the proposed paradigm.

3 Model-Semantics-Aware Process Trace Preprocessing

The computational core of MSA-PPM is *Model-Semantics-Aware Process Trace Preprocessing (MSA-PTP)*, which introduces model-defined semantics between process discovery and predictive learning. Conventional PPM directly maps system-enacted traces to predictive vectors. In contrast, MSA-PTP first interprets and transforms those traces according to semantic knowledge extracted from the corresponding process model. The resulting *model-semantics-aware process traces* are subsequently processed by conventional prefix construction and numerical encoding.

Fig. 1 illustrates this principle. The framework consists of four stages: (1) detection of semantic constructs from the process model, (2) representation of their semantics in a Process Semantics Dictionary (PSD), (3) PSD-guided transformation of system-enacted traces, and (4) conventional predictive encoding. Thus,

$$\mathcal{L} \xrightarrow{\mathcal{D}} M \xrightarrow{\mathcal{E}} PSD \xrightarrow{\mathcal{T}} \widehat{\mathcal{L}} \xrightarrow{\mathcal{P}} X,$$

where $\mathcal{L}$ is the event log, $\mathcal{D}$ denotes process discovery, M is the discovered process model, $\mathcal{E}$ extracts model-defined semantics, $\mathcal{T}$ transforms the traces according to the PSD, and $\mathcal{P}$ performs conventional prefix generation and numerical encoding to produce predictive inputs X.

3.1 Semantic Construct Detection

Let an event log be

$$\mathcal{L} = \{\sigma_1, \sigma_2, \ldots, \sigma_N\},$$

where each system-enacted trace is an ordered event sequence

$$\sigma_i = \langle a_1, a_2, \ldots, a_{|\sigma_i|} \rangle.$$

The observed ordering in σ_i reflects a particular execution but does not necessarily imply a model-defined causal relation between every pair of consecutively observed activities. MSA-PTP therefore first discovers or obtains a process model M and extracts semantic constructs from it before constructing predictive prefixes.

We define semantic extraction as

$$\mathcal{E}(M) = \{\mathcal{E}_{par}(M), \mathcal{E}_{xor}(M), \mathcal{E}_{loop}(M), \mathcal{E}_{sync}(M), \mathcal{E}_{temp}(M), \mathcal{E}_{res}(M), \mathcal{E}_{org}(M), \mathcal{E}_{const}(M)\}.$$

These operators respectively identify parallel, exclusive-choice, loop, synchronization, temporal, resource, organizational, and constraint semantics. The architecture is therefore extensible: additional semantic detectors can be incorporated without changing the subsequent predictive model.

In the implementation evaluated in this paper, M is a Structured Information Control Net (SICN)-oriented process model discovered using the ρ-Algorithm [22, 23]. Its structured sequential, disjunctive, conjunctive, and iterative constructs provide explicit control-flow relations from which the corresponding semantic knowledge can be extracted. In particular, conjunctive constructs provide the parallel sibling relationships required by the parallel-semantics realization developed in Section **??**.

3.2 Process Semantics Dictionary

The *Process Semantics Dictionary (PSD)* provides the intermediate knowledge representation connecting model semantics with trace preprocessing. Rather than embedding semantic knowledge directly into a particular neural architecture, MSA-PTP externalizes it as reusable semantic entries.

The PSD is defined conceptually as

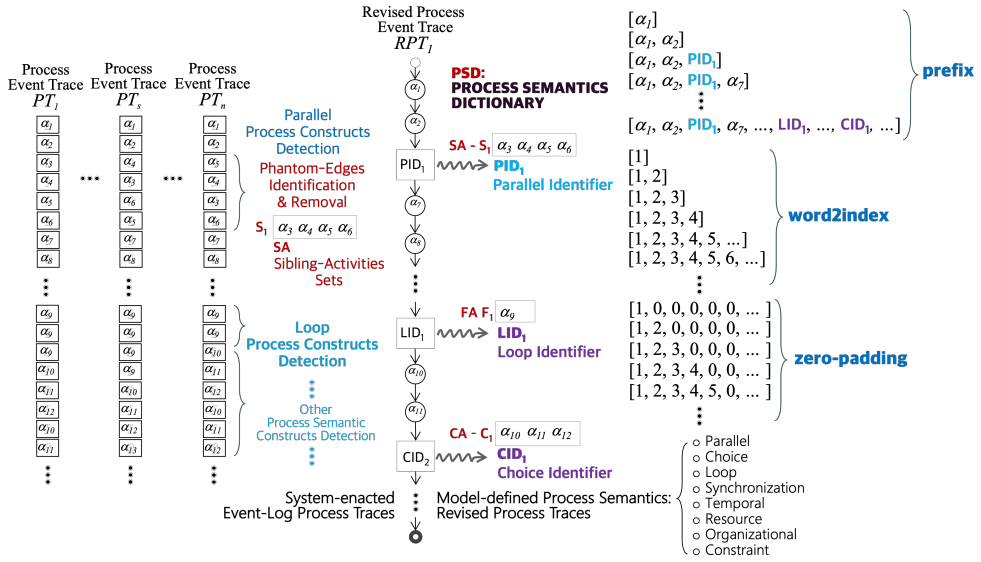


Fig. 1 Conceptual pipeline of Model-Semantics-Aware Process Trace Preprocessing (MSA-PTP).

$$PSD = PSD_{par} \cup PSD_{xor} \cup PSD_{loop} \cup PSD_{sync} \cup PSD_{temp} \cup PSD_{res} \cup PSD_{org} \cup PSD_{const},$$

where each component stores one category of model-defined semantics.

A semantic entry is represented generically as

$$e_k = \langle SID_k, TYPE_k, A_k, R_k, C_k \rangle,$$

where SID_k is a unique semantic identifier, $TYPE_k$ specifies the semantic category, A_k denotes the participating activity set, R_k represents the corresponding behavioral relationships, and C_k optionally contains semantic conditions or contextual constraints.

For example, a parallel-semantic entry can be represented as

$$e_k^{par} = \langle PID_k, PAR, SA_k, R_k^{par} \rangle,$$

where PID_k is a parallel identifier and SA_k is the set of sibling activities belonging to the same parallel construct. Analogously, loop and exclusive-choice constructs may be represented by identifiers such as LID_k and CID_k, as conceptually illustrated in Fig. 1.

This identifier-based representation is important because it converts structural process knowledge into machine-processable semantic knowledge that can be referenced during trace transformation. The PSD therefore functions as an explicit interface between process-model interpretation and predictive-data preparation rather than as an additional feature set tied to a specific learning architecture.

3.3 PSD-Guided Semantic Trace Transformation

The central operation of MSA-PTP is the transformation of a system-enacted trace into a model-semantics-aware trace. Let

$$\sigma = \langle a_1, a_2, \ldots, a_n \rangle$$

denote an observed trace. Its semantic transformation is defined as

$$\widehat{\sigma} = \mathcal{T}_{PSD}(\sigma),$$

with

$$\mathcal{T}_{PSD} : \Sigma^* \to \widehat{\Sigma}^*,$$

where Σ^* and $\widehat{\Sigma}^*$ denote the spaces of conventional and model-semantics-aware traces, respectively.

More generally, the transformation can be expressed as the composition of semantic operators:

$$\mathcal{T}_{PSD} = \mathcal{T}_{const} \circ \mathcal{T}_{org} \circ \mathcal{T}_{res} \circ \mathcal{T}_{temp} \circ \mathcal{T}_{sync} \circ \mathcal{T}_{loop} \circ \mathcal{T}_{xor} \circ \mathcal{T}_{par}.$$

Each operator interprets a trace according to one semantic category represented in the PSD. Depending on the semantics involved, an operator may remove artificial dependencies, identify semantically equivalent executions, preserve iteration or synchronization boundaries, incorporate contextual relationships, or enforce model-defined constraints.

This formulation establishes an important distinction between conventional and model-semantics-aware preprocessing:

$$\underbrace{\sigma \xrightarrow{\mathcal{P}} X}_{\text{Conventional PPM}} \qquad \Longrightarrow \qquad \underbrace{\sigma \xrightarrow{\mathcal{T}_{PSD}} \widehat{\sigma} \xrightarrow{\mathcal{P}} \widehat{X}}_{\text{MSA-PPM}}.$$

Thus, MSA-PPM changes the *semantic content of the predictive input* before numerical encoding rather than requiring modification of the predictive architecture itself.

3.4 Operational Realization through Parallel Semantics

The complete PSD defines an extensible semantic framework; however, this study operationalizes and experimentally evaluates its *parallel-semantics* component. Accordingly, the transformation used in the present experiments is

$$\widehat{\sigma} = \mathcal{T}_{par}(\sigma, PSD_{par}).$$

Let

$$SA_k = \{a_{k1}, a_{k2}, \ldots, a_{km}\}$$

denote the sibling-activity set associated with a parallel identifier PID_k. Activities in SA_k belong to the same model-defined parallel construct and therefore need not

possess causal precedence merely because they appear consecutively in a particular execution trace.

For two activities $a_i, a_j \in SA_k$, an observed directly-following relation

$$a_i >_\sigma a_j$$

does not, by itself, imply

$$a_i \rightarrow_M a_j,$$

where $\rightarrow_M$ denotes model-defined causal precedence. When

$$a_i >_\sigma a_j \quad \wedge \quad a_i \parallel_M a_j,$$

the directed relation appearing in the sequential trace is treated as a *phantom edge*: an execution-order artifact that is present in the system-enacted trace but absent from the model-defined causal structure.

The parallel-semantic transformation therefore uses PSD_{par} to identify sibling activities, eliminate their artificial directed dependencies, and represent their shared parallel semantics through the corresponding PID_k, as illustrated in Fig. 1. This produces a model-semantics-aware trace in which arbitrary parallel interleavings no longer impose false sequential dependencies on predictive learning.

After semantic transformation, conventional preprocessing remains unchanged:

$$\widehat{\sigma} \rightarrow \text{prefixes} \rightarrow \texttt{word2index} \rightarrow \text{zero-padding} \rightarrow \widehat{X}.$$

Consequently, the same semantically revised predictive inputs can be supplied to different learning architectures. This study evaluates this property using LSTM- and Transformer-based next-activity prediction models. The detailed construction of PSD_{par} and the corresponding Phantom-Edge-Aware Semantic Preprocessing algorithm are presented in the following section.

4 Next-Activity Prediction: A Case Study of the Proposed MSA-PPM Paradigm

The proposed Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM) paradigm is designed as a general semantic preprocessing framework applicable to different predictive tasks and learning architectures. To demonstrate its operational effectiveness, this study considers *next-activity prediction* as a representative PPM task, where the prediction of an immediate successor activity is particularly sensitive to the behavioral relationships encoded in process traces. Specifically, the parallel-semantics realization of MSA-PPM developed in the previous section is integrated into a conventional next-activity prediction pipeline. Model-defined parallel semantics are used to transform system-enacted traces into model-semantics-aware traces before prefix construction and numerical encoding. The resulting predictive inputs are then supplied to representative LSTM- and Transformer-based models without modifying their original learning architectures.

The objective of this case study is therefore not to introduce a new predictive architecture, but to evaluate whether *semantic preprocessing itself* can improve

next-activity prediction. By comparing conventional log-centric preprocessing with the proposed model-semantics-aware preprocessing under otherwise equivalent predictive settings, the experiments isolate the performance contribution of model-defined parallel semantics.

4.1 Preprocessing for Next-Activity Prediction

This case study operationalizes the parallel-semantics component of MSA-PPM for next-activity prediction. The objective is to determine whether model-defined parallel semantics can improve predictive inputs without modifying the underlying learning architecture. To isolate this effect, the same LSTM- and Transformer-based prediction models are trained using conventional and model-semantics-aware inputs, respectively.

The proposed realization consists of three steps: (1) detection of parallel process constructs and their sibling-activity sets, (2) identification of directed phantom edges and construction of the parallel-semantics PSD, and (3) generation of Model-Semantics-Aware Prefix (MSAP) vectors for predictive learning.

4.1.1 Parallel Process Construct Detection

The first step extracts model-defined parallel semantics from the discovered process model. Let

$$M = (A, G, F)$$

denote the SICN-oriented process model discovered by the ρ-Algorithm, where A is the set of activities, G the set of routing gateways, and F the set of model-defined control-flow relations. The ρ-Algorithm identifies structured process patterns, including conjunctive constructs representing parallel execution.

For each discovered conjunctive construct, a sibling-activity set is defined as

$$SA_k = \{a_{k1}, a_{k2}, \ldots, a_{km}\},$$

where all activities in SA_k belong to the same parallel block. Because sibling activities are concurrently enabled within the corresponding construct, their observed ordering in an individual trace does not, by itself, constitute a causal dependency.

Fig. 2 illustrates this distinction between an observed sequential relation and model-defined parallel semantics. The discovered sibling-activity sets provide the semantic knowledge required to determine whether an adjacency observed in an event trace represents a genuine model-defined dependency or an execution-order artifact.

4.1.2 Directed Phantom-Edge Identification

Let

$$\sigma = \langle a_1, a_2, \ldots, a_n \rangle$$

be a system-enacted process trace, and let $a_i \succ_\sigma a_j$ denote that a_j directly follows a_i in σ. Conventional sequence-based preprocessing preserves this observed ordering and may therefore present it to a predictive model as a sequential dependency.

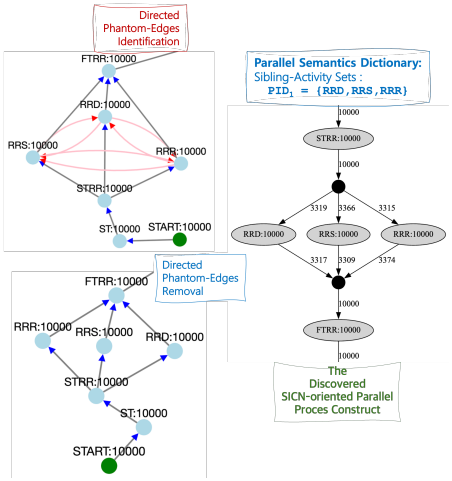


Fig. 2 Directed phantom edges induced by arbitrary execution orders among parallel sibling activities.

For parallel sibling activities, however, such an ordering may be incidental. A directed relation (a_i, a_j) is defined as a *phantom edge* when

$$a_i \succ_\sigma a_j, \qquad \exists SA_k : a_i, a_j \in SA_k, \qquad (a_i, a_j) \notin F.$$

Thus, the relation is observed in the system-enacted trace but is not a genuine causal relation in the discovered process model.

Accordingly, the set of directed phantom edges is

$$PE = \{(a_i, a_j) \mid a_i \succ_\sigma a_j,\ \exists SA_k : \{a_i, a_j\} \subseteq SA_k,\ (a_i, a_j) \notin F\}.$$

Algorithm 1 identifies these relations by examining adjacent activity pairs in each trace and testing their membership in the discovered sibling-activity sets.

Algorithm 1 Directed Phantom-Edge Identification

Require: Event log $L = \{\sigma_1, \ldots, \sigma_N\}$
Require: Model-defined control-flow relations F
Require: Sibling-activity sets $\mathcal{SA} = \{SA_1, \ldots, SA_r\}$
Ensure: Directed phantom-edge set PE

$PE \leftarrow \emptyset$
for each trace $\sigma \in L$ **do**
  for each adjacent pair (a_i, a_{i+1}) in σ **do**
    if $\exists SA_k \in \mathcal{SA}$ such that $a_i, a_{i+1} \in SA_k$ **then**
      if $(a_i, a_{i+1}) \notin F$ **then**
        $PE \leftarrow PE \cup \{(a_i, a_{i+1})\}$
      end if
    end if
  end for
end for
return PE

This identification provides the explicit connection between the system-enacted trace and its model-defined interpretation: observed adjacency is preserved only when it represents a genuine model-defined dependency; otherwise, its artificial directional meaning is removed through the semantic abstraction described below.

4.1.3 Parallel-Semantics Dictionary and Trace Transformation

The discovered sibling-activity sets are represented in the parallel-semantics component of the Process Semantics Dictionary. For this case study, it is defined as

$$PSD_{par} = \{PID_1 \mapsto SA_1,\ PID_2 \mapsto SA_2,\ \ldots,\ PID_r \mapsto SA_r\},$$

where PID_k is the semantic identifier associated with sibling-activity set SA_k.

Definition 1. ***Parallel-Semantics PSD Entry.*** *A parallel-semantics entry*

$$PID_k \mapsto SA_k$$

associates a model-defined parallel construct with its sibling-activity set. Activities belonging to SA_k share the same parallel semantic context and are represented by PID_k during semantic trace transformation.

Let

$$\phi_{par} : A \to A \cup PID$$

denote the parallel-semantic mapping function:

$$\phi_{par}(a) = \begin{cases} PID_k, & a \in SA_k, \\ a, & \text{otherwise.} \end{cases}$$

Applying ϕ_{par} to a system-enacted trace removes the predictive significance of arbitrary ordering among activities belonging to the same parallel construct. Consequently, traces that differ only in the execution order of the same parallel siblings are mapped to a common semantic representation.

For example,

$$\langle a_p, a_q, a_r \rangle \quad \text{and} \quad \langle a_q, a_p, a_r \rangle, \qquad a_p, a_q \in SA_k,$$

contain different system-enacted orders but equivalent parallel semantics. Through PSD_{par}, both are interpreted using the common semantic identifier PID_k, thereby preventing their arbitrary ordering from being learned as a genuine causal dependency.

Fig. 3 summarizes this operational pipeline: the ρ-Algorithm discovers the SICN-oriented process model, parallel constructs are represented as PSD_{par} entries, and the corresponding semantic transformation produces model-semantics-aware traces before predictive encoding.

4.1.4 Model-Semantics-Aware Prefix Vector Construction

The transformed traces are finally converted into fixed-length predictive inputs. Let

$$\widehat{\sigma} = \mathcal{T}_{par}(\sigma, PSD_{par}) = \langle \hat{a}_1, \hat{a}_2, \ldots, \hat{a}_{|\widehat{\sigma}|} \rangle$$

denote a parallel-semantics-aware trace. For prediction point k, its semantic prefix is

$$\widehat{\sigma}^{(k)} = \langle \hat{a}_1, \hat{a}_2, \ldots, \hat{a}_k \rangle.$$

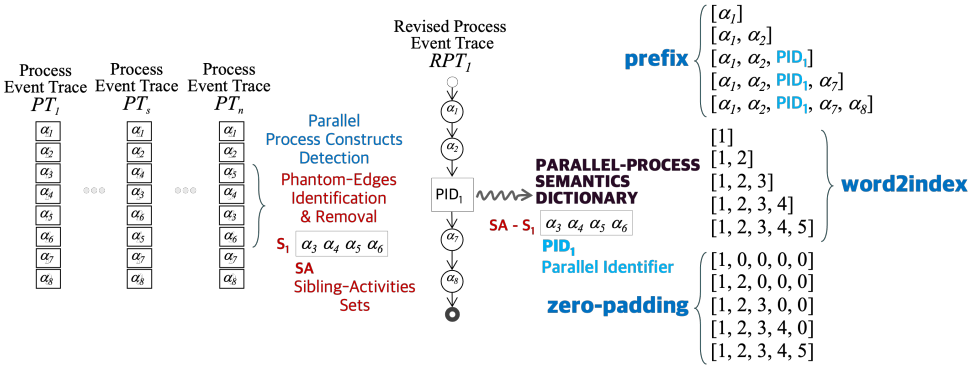


Fig. 3 Construction of Model-Semantics-Aware Prefix (MSAP) vectors using parallel semantics.

Definition 2. ***Model-Semantics-Aware Prefix (MSAP) Vector.*** *An MSAP vector is a fixed-length numerical representation of a predictive prefix generated after model-defined semantic transformation. In the present realization, parallel sibling activities are interpreted according to PSD_{par} so that arbitrary execution-order relations among them are not retained as directed dependencies in the predictive representation.*

After semantic transformation, the remaining preprocessing follows the conventional PPM pipeline:

$$\sigma \xrightarrow{\mathcal{T}_{par}} \widehat{\sigma} \to \widehat{\sigma}^{(k)} \to \texttt{word2index} \to \text{zero-padding} \to MSAP_k.$$

If $L_{\max}$ denotes the maximum input length, the resulting vector is

$$MSAP_k = [x_1, x_2, \ldots, x_k, PAD, \ldots, PAD],$$

with total length $L_{\max}$. Each semantic identifier and ordinary activity identifier is subsequently mapped to the same embedding space used by the downstream prediction model.

The proposed method therefore changes the *semantic composition* of the predictive input while leaving the downstream learning procedure unchanged. This property enables a controlled comparison between conventional and model-semantics-aware preprocessing using the same LSTM and Transformer architectures.

4.1.5 Computational Considerations

The proposed preprocessing consists of an offline process-discovery phase and a trace-transformation phase. Process discovery using the ρ-Algorithm is performed once for the event log. During trace transformation, each event is examined against the parallel-semantics PSD to determine its semantic mapping. With indexed PSD membership, this transformation is linear in the total number of events, i.e., $O(|E_L|)$, where $|E_L|$ denotes the number of recorded events.

Accordingly, MSA-PTP introduces preprocessing overhead without changing the computational structure of subsequent model training or inference. Its experimental value can therefore be evaluated independently of modifications to the predictive architectures.

4.2 Next-Activity Prediction and Performance Analysis

This subsection evaluates the predictive effect of the proposed model-semantics-aware preprocessing through next-activity prediction. The experimental framework consists of two consecutive phases: *semantic preprocessing* and *predictive learning*. In the first phase, the parallel-semantics-aware preprocessing described in the preceding subsection transforms system-enacted process traces into model-semantics-aware traces by resolving directed phantom edges associated with arbitrary execution orders among parallel sibling activities. The transformed traces are then encoded as fixed-length Model-Semantics-Aware Prefix (MSAP) vectors using the parallel-semantics PSD and conventional numerical encoding and zero-padding procedures.

In the second phase, the resulting MSAP vectors are supplied to existing next-activity prediction models without modifying their network architectures. This separation enables the predictive effect of semantic preprocessing to be evaluated independently of architectural changes. To examine whether the observed effect is consistent across different sequence-learning mechanisms, this study employs two representative architectures: Long Short-Term Memory (LSTM), representing recurrent sequence learning, and Transformer, representing self-attention-based sequence learning.

Accordingly, the experimental comparison evaluates

$$\text{Conventional Prefixes} \rightarrow \text{LSTM/Transformer}$$

against

$$\text{MSAP Vectors} \rightarrow \text{LSTM/Transformer},$$

while maintaining equivalent predictive architectures and experimental settings. The resulting performance differences therefore provide direct evidence of the contribution of model-defined parallel semantics to next-activity prediction.

4.2.1 Experimental Setup

(1) Data Setup: Large Bank Transaction Dataset

To evaluate the effectiveness of the proposed Model-Semantics-Aware Prefix (MSAP) preprocessing methodology, a series of next-activity prediction experiments were conducted using the Large Bank Transaction Event Log Dataset [28], which is publicly available from the 4TU Research Data Repository. The dataset is a synthetic benchmark specifically designed for research on business process management, process mining, and predictive process monitoring. It was generated from a Petri-net-based business process model and contains diverse execution patterns together with realistic routing variations and execution noise. The dataset consists of 10,000 process

instances, 678,864 recorded events, and 113 distinct activity types, providing a sufficiently large and structurally diverse benchmark for evaluating predictive process monitoring models.

(2) Computing Environment Setup

Here, we summarize the experimental computing and implementing environments:

- All experiments were performed on a workstation equipped with an AMD Ryzen 7 5800X 8-Core Processor, an NVIDIA RTX 3070 GPU, and 32 GB of main memory running Ubuntu 20.04.6.
- The experimental programs were implemented in Python 3.9.19 using PyTorch 1.12.1 and TensorFlow 2.11.0.
- Specifically, the LSTM prediction model was implemented using PyTorch, whereas the Transformer-based prediction model was implemented using TensorFlow.

(3) MSAP Preprocessing Setup: Parallel-Process Semantics Dictionary and Slibling-Activity Sets

Before constructing the Model-Semantics-Aware Prefix (MSAP) vectors, the proposed preprocessing methodology first identifies the parallel execution semantics embedded in the event log. This preprocessing stage constitutes the semantic foundation of the proposed predictive process monitoring framework because the subsequent MSAP vector construction relies on accurate identification of sibling activities belonging to the same parallel control-flow construct.

To this end, the prepared Large Bank Transaction Event Log Dataset was analyzed using the ρ-Algorithm-based process mining system. The system first identifies and removes directed phantom edges generated by arbitrary execution orders among sibling activities and subsequently discovers the exact SICN-oriented business process model. Based on the discovered process model, all conjunctive process patterns are extracted together with their corresponding sibling-activity sets. These sibling-activity sets are then organized into a Parallel-Process Semantics Dictionary, which serves as the semantic knowledge base for the proposed MSAP preprocessing methodology.

Fig. 4 illustrates the construction results of the Parallel-Process Semantics Dictionary. The upper-part in the figure presents the phantom-edge identification and elimination procedure performed during process discovery, while the lower-part visualizes the discovered parallel process constructs together with their corresponding sibling-activity sets. As shown in the figure, the experimental dataset contains 12 single-level parallel control-flow constructs, each represented by a unique Parallel-Process Semantics Dictionary entry denoted as $(SA_1, SA_2, \ldots, SA_{12})$. Each entry corresponds to a complete sibling-activity set associated with a single conjunctive process pattern and provides a unique semantic identifier for all activities belonging to that parallel construct.

During MSAP vector construction, every activity contained in a sibling-activity set is replaced by the identifier of its corresponding entry in Parallel-Process Semantics Dictionary, denoted as $(PID_1, PID_2, \ldots, PID_{12})$. All the process traces that differ only in the execution order of sibling activities are transformed into identical semantic representations before zero-padding is applied. This dictionary-based semantic abstraction eliminates artificial sequential dependencies while preserving the genuine

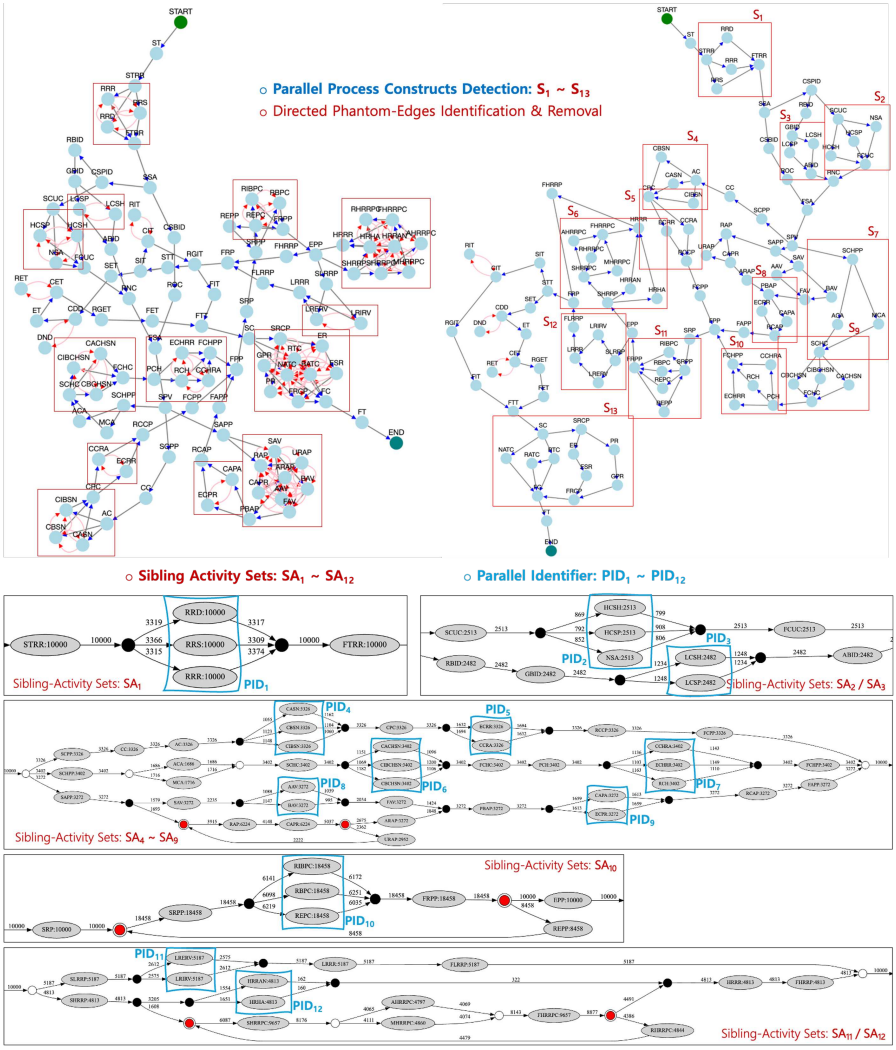


Fig. 4 Phatom-Edge Identification/Removal and Sibling-Activity Sets on the Large Bank Transaction Process Dataset

causal relationships defined by the discovered SICN process model. As a result, the generated MSAP vectors provide semantically consistent input representations for both the LSTM- and Transformer-based next-activity prediction models evaluated in this case study.

4.2.2 Performance Evaluation Setup

The predictive performance of the proposed methodology was evaluated using four widely adopted performance measures for multi-class classification problems, namely

Accuracy, Precision, Recall, and F1-score. Accuracy measures the proportion of correctly predicted activities among all prediction instances. Precision evaluates the proportion of correctly predicted activities among the activities predicted by the model. Recall measures the proportion of actual next activities that are correctly identified by the prediction model. The F1-score, defined as the harmonic mean of Precision and Recall, provides a balanced evaluation of prediction performance, particularly when class distributions are uneven.

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)} \tag{1}$$

$$Precision = \frac{(TP)}{(TP + FP)} \tag{2}$$

$$Recall = \frac{(TP)}{(TP + FN)} \tag{3}$$

$$F1 = 2 \times \frac{(Precision \times Recall)}{(Precision + Recall)} \tag{4}$$

Let (TP), (FP), (TN), and (FN) denote the numbers of true positives, false positives, true negatives, and false negatives, respectively. The four evaluation metrics are computed using Equations (1)-(4). Higher values of these metrics indicate superior prediction performance and are therefore used to quantitatively compare the effectiveness of the conventional prefix representation and the proposed Model-Semantics-Aware Prefix (MSAP) representation.

To ensure a fair comparison, all prediction models were trained and evaluated under identical experimental conditions. The only difference between the baseline models and the proposed models was the preprocessing methodology used to construct the input prefix vectors. Any observed performance improvement, therefore, can be directly attributed to the proposed MSAP preprocessing rather than to differences in network architectures or training configurations.

4.2.3 MSA-PPM-LSTM: Incremental Semantic Validation through Next-Activity Prediction

To provide the first empirical validation of the proposed Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM) paradigm, we conducted a series of next-activity prediction experiments using a Long Short-Term Memory (LSTM)-based predictive process monitoring model [2, 12]. LSTM networks have been widely adopted in predictive process monitoring because of their capability to capture long-term temporal dependencies from sequential event prefixes [29]. In the present case study, the LSTM architecture serves as the baseline prediction model for evaluating the effectiveness of the proposed MSA-PPM semantic preprocessing framework.

Within the proposed framework, the semantically revised process traces generated by the MSA-PPM preprocessing pipeline are transformed into predictive prefixes and subsequently encoded into embedding vectors. These embedded representations are sequentially processed by the LSTM network to construct hidden representations describing the execution context of partially completed process instances. The hidden

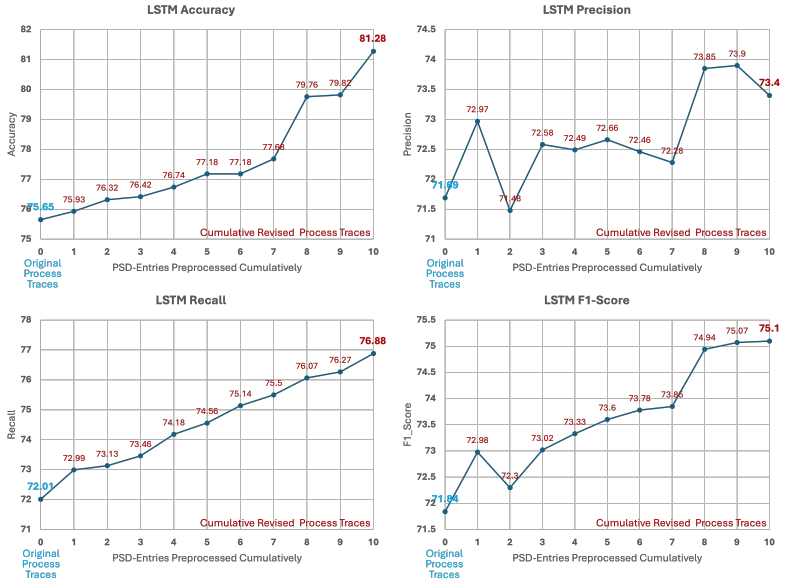


Fig. 5 Evolution of Predictive Performance during Cumulative Parallel-Semantics Preprocessing

Table 1 Incremental Performance Improvement of the MSA-PPM-LSTM Model through Cumulative Parallel-Semantics Preprocessing

Cumulative PSD-Entry (PID) Preprocessed	Accuracy	Precision	Recall	F1-Score
`Original Process Traces`	**75.65**	**71.69**	**72.01**	**71.84**
PID_1 `MSAP-Revised Process Traces`	75.93	72.97	72.99	72.98
PID_2 `MSAP-Revised Process Traces`	76.32	71.48	73.13	72.30
PID_3 `MSAP-Revised Process Traces`	76.42	72.58	73.46	73.02
PID_4 `MSAP-Revised Process Traces`	76.74	72.49	74.18	73.33
PID_5 `MSAP-Revised Process Traces`	77.18	72.66	74.56	73.60
PID_6 `MSAP-Revised Process Traces`	77.18	72.46	75.14	73.78
PID_7 `MSAP-Revised Process Traces`	77.68	72.28	75.50	73.85
PID_8 `MSAP-Revised Process Traces`	79.76	73.85	76.07	74.94
PID_9 `MSAP-Revised Process Traces`	79.82	73.90	76.27	75.07
PID_{10} `MSAP-Revised Process Traces`	**81.28**	**73.40**	**76.88**	**75.10**

Table 2 Overall Performance Improvement of the Proposed MSA-PPM-LSTM Model

Metric	Baseline-LSTM	MSA-PPM-LSTM	Improvement
Accuracy	75.65	81.28	+**5.63**
Precision	71.69	73.40	+**1.71**
Recall	72.01	76.88	+**4.87**
F1-score	71.84	75.10	+**3.26**

representation produced at the final time step is then connected to a fully connected layer followed by a Softmax classifier to estimate the probability distribution of the next activity.

It is important to emphasize that the proposed MSA-PPM-LSTM model does not modify the internal architecture of the LSTM network. Instead, the proposed paradigm improves predictive performance exclusively through semantic refinement of the input process traces before conventional predictive learning is performed. To ensure a fair comparison, the Baseline-LSTM and MSA-PPM-LSTM models employ exactly the same network architecture, hyperparameter settings, optimization strategy, and training procedure. The Baseline-LSTM model is trained using conventional zero-padded activity prefixes generated directly from the original event log, whereas the MSA-PPM-LSTM model is trained using the semantically revised process traces produced by the proposed MSA-PPM preprocessing framework. Lastly, semantic preprocessing constitutes the only experimental variable, allowing the contribution of the proposed MSA-PPM paradigm to be evaluated independently of the prediction architecture.

The experiments were conducted using the Large Bank Transaction Event Log Dataset described in the previous subsection. The semantically revised process traces were divided into training and testing datasets using an 80:20 ratio. Both prediction models were trained for 100 epochs using the Adam optimizer with a learning rate of 0.001 and a batch size of 12. To investigate how progressively incorporating model-defined parallel semantics influences predictive performance, we designed an incremental semantic validation experiment. Rather than applying all discovered parallel-process semantics simultaneously, semantic preprocessing was performed cumulatively according to the entries stored in the Parallel Process Semantics Dictionary (PSD). Beginning with the original process traces, each PSD entry, corresponding to one discovered parallel process construct, was sequentially applied to eliminate the directed phantom edges associated with that construct. After each preprocessing step, the revised process traces were accumulated and subsequently used to train and evaluate the same LSTM prediction model.

This experimental design enables both the individual and cumulative contributions of model-defined parallel semantics to be quantitatively analyzed. Since neither the prediction architecture nor the training procedure changes throughout the experiment, the observed performance variations directly reflect the effectiveness of the proposed semantic preprocessing methodology. Table 1 and Fig. 5 together illustrate the evolution of predictive performance during cumulative semantic preprocessing. The first row of Table 1 corresponds to the original process traces without semantic preprocessing and serves as the experimental baseline. Each subsequent row represents the accumulated revised process traces obtained after incorporating one additional PSD entry. Fig. 5 graphically depicts the same experimental results and clearly reveals the progressive improvement achieved as increasing amounts of model-defined parallel semantics are incorporated into the predictive process traces.

Several significant observations can be drawn from these results. First, all four evaluation metrics exhibit a consistent upward trend throughout the cumulative preprocessing process, demonstrating that the proposed MSA-PPM framework continuously improves the semantic quality of predictive process traces. Second, the

improvements are progressive rather than abrupt, indicating that each PSD entry contributes complementary semantic knowledge toward refining the predictive representation. Third, relatively larger performance gains are observed after incorporating the later PSD entries, particularly from PID_8 onward, suggesting that certain parallel process constructs contribute more substantially to eliminating execution-order ambiguities introduced by directed phantom edges.

Overall, prediction accuracy increases monotonically from 75.65% for the original process traces to 81.28% after all ten PSD entries have been incorporated, representing an absolute improvement of 5.63 percentage points. Similar improvements are consistently observed for Precision, Recall, and F1-score, confirming that progressively incorporating model-defined parallel semantics systematically enhances both the correctness and robustness of next-activity prediction. Table 2 summarizes the overall performance improvement achieved after incorporating the complete Parallel Process Semantics Dictionary into the proposed MSA-PPM preprocessing framework. Compared with the Baseline-LSTM model, the proposed MSA-PPM-LSTM consistently achieves superior performance across all evaluation metrics. In particular, prediction accuracy improves by 5.63 percentage points, while the F1-score increases by 3.26 percentage points, demonstrating that semantic refinement simultaneously improves predictive precision and predictive completeness.

These experimental results provide the first empirical validation of the proposed MSA-PPM paradigm. The observed performance improvements originate from semantic refinement rather than architectural enhancement. By progressively incorporating model-defined parallel-process semantics into predictive process traces, the proposed MSA-PPM framework enables conventional LSTM models to learn genuine behavioral dependencies while eliminating execution-order artifacts introduced by directed phantom edges. Consequently, the study demonstrates that Model-Semantics-Aware Process Trace Preprocessing constitutes an effective, architecture-independent strategy for improving deep learning-based predictive process monitoring. Furthermore, the cumulative semantic validation confirms that the Process Semantics Dictionary functions as an extensible semantic knowledge base whose incremental semantic enrichment directly translates into progressively improved predictive performance.

4.2.4 MSA-PPM-Transformer: Incremental Semantic Validation through Next-Activity Prediction

To provide a second empirical validation of the proposed Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM) paradigm, we conducted an additional series of next-activity prediction experiments using a Transformer-based predictive process monitoring model [1, 13, 30]. Transformer architectures have recently demonstrated outstanding performance in predictive process monitoring owing to their multi-head self-attention mechanism, which effectively captures both local and global contextual dependencies among process activities [31, 32]. In the present case study, the Transformer serves as a representative attention-based prediction model for evaluating the architecture-independent applicability of the proposed MSA-PPM semantic preprocessing framework.

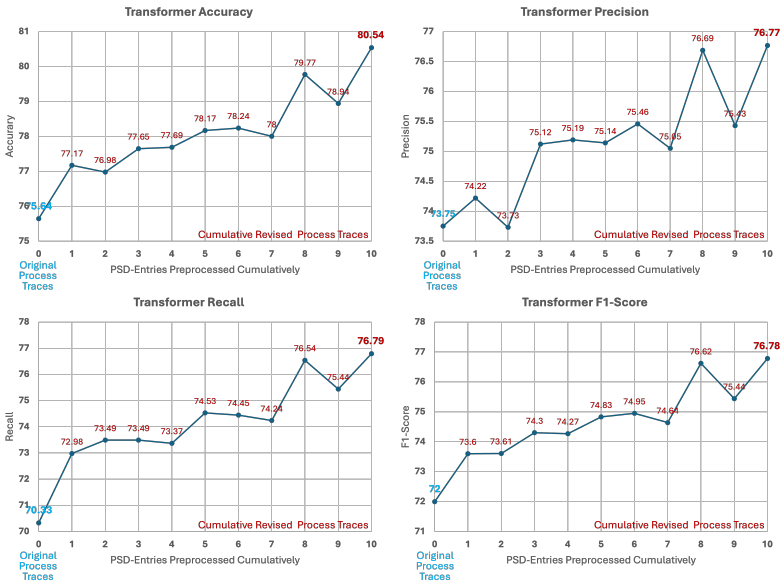


Fig. 6 Evolution of Predictive Performance during Cumulative Parallel-Semantics Preprocessing

Within the proposed framework, the semantically revised process traces generated by the MSA-PPM preprocessing pipeline are transformed into predictive prefixes and encoded into dense activity embeddings together with positional embeddings before being supplied to the Transformer encoder. Through multiple self-attention layers, the Transformer learns contextual relationships among process activities and generates high-level semantic representations of partially executed process instances. The resulting representations are subsequently connected to a fully connected prediction layer followed by a Softmax classifier to estimate the probability distribution of the next activity.

It is important to emphasize that the proposed MSA-PPM-Transformer model does not modify the internal architecture of the Transformer network. Instead, predictive performance is improved exclusively through semantic refinement of the input process traces before conventional Transformer-based predictive learning is performed. To ensure a fair comparison, the Baseline-Transformer and MSA-PPM-Transformer models employ exactly the same Transformer architecture, hyperparameter settings, optimization strategy, and training procedure. The Baseline-Transformer model is trained using conventional zero-padded activity prefixes generated directly from the original event log, whereas the MSA-PPM-Transformer model is trained using the semantically revised process traces produced by the proposed MSA-PPM preprocessing framework. Therefore, semantic preprocessing constitutes the only experimental variable, allowing the contribution of the proposed MSA-PPM paradigm to be evaluated independently of the prediction architecture.

The experiments were conducted using the same Large Bank Transaction Event Log Dataset and the same experimental protocol adopted in the previous LSTM experiments. The semantically revised process traces were divided into training and

Table 3 Incremental Performance Improvement of the MSA-PPM-Transformer Model through Cumulative Parallel-Semantics Preprocessing

Cumulative PSD-Entry (PID) Preprocessed	Accuracy	Precision	Recall	F1-Score
`Original Process Traces`	**75.64**	**73.75**	**70.33**	**72.00**
PID_1 `MSAP-Revised Process Traces`	77.17	74.22	72.98	73.60
PID_2 `MSAP-Revised Process Traces`	76.98	73.73	73.49	73.61
PID_3 `MSAP-Revised Process Traces`	77.65	75.12	73.49	74.30
PID_4 `MSAP-Revised Process Traces`	77.69	75.19	73.37	74.27
PID_5 `MSAP-Revised Process Traces`	78.17	75.14	74.53	74.83
PID_6 `MSAP-Revised Process Traces`	78.24	75.46	74.45	74.95
PID_7 `MSAP-Revised Process Traces`	78.00	75.05	74.24	74.64
PID_8 `MSAP-Revised Process Traces`	79.77	76.69	76.54	76.62
PID_9 `MSAP-Revised Process Traces`	78.94	75.43	75.44	75.44
PID_{10} `MSAP-Revised Process Traces`	**80.54**	**76.77**	**76.79**	**76.78**

Table 4 Overall Performance Improvement of the Proposed MSA-PPM-Transformer Model

Metric	Baseline-Transformer	MSA-PPM-Transformer	Improvement
Accuracy	75.64	80.54	+**4.90**
Precision	73.75	76.77	+**3.02**
Recall	70.33	76.79	+**6.46**
F1-score	72.00	76.78	+**4.78**

testing datasets using an 80:20 ratio. Both Transformer models were trained for 100 epochs using the Adam optimizer with a learning rate of 0.001 and a batch size of 12. To investigate whether the cumulative incorporation of model-defined parallel semantics exhibits similar behavior under an attention-based prediction architecture, the same incremental semantic validation experiment described in the previous subsection was repeated using the Transformer model. Rather than incorporating all discovered parallel-process semantics simultaneously, semantic preprocessing was performed cumulatively according to the entries stored in the Parallel Process Semantics Dictionary (PSD). Beginning with the original process traces, each PSD entry, corresponding to one discovered parallel process construct, was sequentially applied to eliminate the directed phantom edges associated with that construct. After each preprocessing step, the revised process traces were accumulated and subsequently used to train and evaluate the same Transformer prediction model.

This experimental design enables both the individual and cumulative contributions of model-defined parallel semantics to be quantitatively analyzed using an attention-based neural architecture. Since neither the Transformer architecture nor the training procedure changes throughout the experiment, the observed performance variations directly reflect the effectiveness of the proposed semantic preprocessing methodology. Table 3 and Fig. 6 together illustrate the evolution of predictive performance during cumulative semantic preprocessing. The first row of Table 3 corresponds to the original process traces without semantic preprocessing and serves as the experimental baseline. Each subsequent row represents the accumulated revised process traces

obtained after incorporating one additional PSD entry. Fig. 6 graphically depicts the same experimental results and clearly illustrates the progressive improvement achieved as increasing amounts of model-defined parallel semantics are incorporated into the predictive process traces.

Several important observations can be drawn from these results. First, all four evaluation metrics exhibit an overall upward trend throughout the cumulative semantic preprocessing process, demonstrating that the proposed MSA-PPM framework consistently improves predictive performance under the Transformer architecture. Second, the performance improvements are progressive rather than abrupt, indicating that each PSD entry contributes complementary semantic knowledge toward refining the predictive representation. Third, similar to the LSTM experiments, relatively larger performance gains are observed after incorporating the later PSD entries, particularly from PID_8 onward, suggesting that several parallel process constructs contribute more substantially to eliminating execution-order ambiguities introduced by directed phantom edges.

Overall, prediction accuracy increases from 75.64% for the original process traces to 80.54% after incorporating all ten PSD entries, corresponding to an absolute improvement of 4.90 percentage points. More importantly, Recall and F1-score exhibit even larger improvements, increasing by 6.46 and 4.78 percentage points, respectively. These results indicate that semantic refinement substantially enhances the Transformer's ability to correctly identify future activities while simultaneously improving the overall balance between prediction precision and prediction completeness.

Table 4 summarizes the overall performance improvement achieved after incorporating the complete Parallel Process Semantics Dictionary into the proposed MSA-PPM preprocessing framework. Compared with the Baseline-Transformer model, the proposed MSA-PPM-Transformer consistently achieves superior performance across all evaluation metrics. In particular, prediction accuracy improves by 4.90 percentage points, while the F1-score increases by 4.78 percentage points, demonstrating that semantic refinement simultaneously improves predictive accuracy, predictive robustness, and class-balanced prediction performance.

These experimental results provide a second empirical validation of the proposed MSA-PPM paradigm using an attention-based prediction architecture. Similar to the LSTM experiments, the observed performance improvements originate exclusively from semantic refinement rather than architectural enhancement. By progressively incorporating model-defined parallel-process semantics into predictive process traces, the proposed MSA-PPM framework enables the Transformer model to learn genuine behavioral dependencies while suppressing execution-order artifacts introduced by directed phantom edges. Moreover, the consistency of the experimental results obtained from both recurrent and attention-based neural architectures demonstrates that the proposed MSA-PPM framework constitutes an architecture-independent semantic preprocessing paradigm for predictive process monitoring. Furthermore, the cumulative semantic validation again confirms that the Process Semantics Dictionary functions as an extensible semantic knowledge base whose progressive semantic enrichment directly translates into progressively improved predictive performance.

4.2.5 Summary: Architecture-Independent Validation of the Proposed MSA-PPM Paradigm

Table 5 presents an overall comparison of the experimental results obtained from the two representative deep learning architectures employed in this study, namely the recurrent LSTM model and the attention-based Transformer model. The objective of this comparison is not to determine which prediction architecture performs better, but rather to investigate whether the proposed Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM) framework consistently improves predictive performance regardless of the underlying deep learning model. Since the Baseline and MSA-PPM models employ identical prediction architectures, hyperparameter settings, optimization strategies, and experimental protocols, semantic preprocessing constitutes the only experimental variable. Accordingly, the observed performance improvements can be attributed exclusively to the proposed MSA-PPM framework rather than to modifications of the underlying deep learning architectures.

For the LSTM-based prediction model, the proposed MSA-PPM framework improves prediction accuracy from 75.65% to 81.28%, corresponding to an absolute improvement of 5.63 percentage points. Likewise, the F1-score increases from 71.84% to 75.10%, representing an improvement of 3.26 percentage points. These improvements indicate that semantic refinement enables the recurrent neural network to learn genuine behavioral dependencies more effectively by eliminating artificial sequential relationships introduced by directed phantom edges.

Table 5 Architecture-Independent Performance Validation of the Proposed MSA-PPM Paradigm

Model	Accuracy	ΔAccuracy	F1-Score	ΔF1-Score
Baseline-LSTM	75.65	–	71.84	–
MSA-PPM-LSTM	**81.28**	**+5.63**	**75.10**	**+3.26**
Baseline-Transformer	75.64	–	72.00	–
MSA-PPM-Transformer	**80.54**	**+4.90**	**76.78**	**+4.78**

A similar tendency is observed for the Transformer-based prediction model. Prediction accuracy increases from 75.64% to 80.54%, corresponding to an absolute improvement of 4.90 percentage points, while the F1-score increases from 72.00% to 76.78%, representing an improvement of 4.78 percentage points. In particular, the substantial improvement in Recall demonstrates that semantic refinement significantly enhances the Transformer's ability to correctly identify subsequent activities while preserving prediction precision.

More importantly, both prediction models exhibit remarkably consistent performance improvements despite employing fundamentally different sequence-learning mechanisms. Whereas the LSTM model captures temporal dependencies through recurrent memory cells, the Transformer model learns contextual relationships using multi-head self-attention. Nevertheless, both architectures consistently benefit from the proposed semantic preprocessing, demonstrating that the effectiveness of the MSA-PPM framework is independent of the internal learning mechanism of the prediction model.

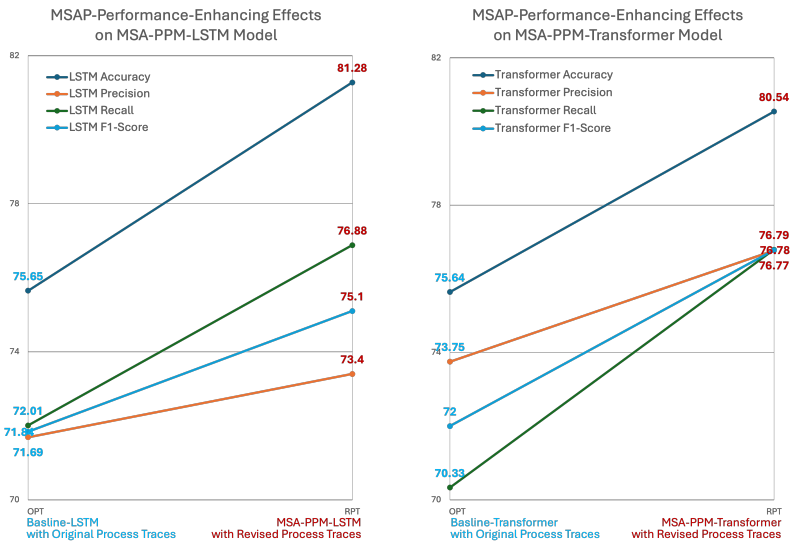


Fig. 7 Architecture-Independent Performance Improvements Achieved by the Proposed MSA-PPM Framework

The cumulative semantic validation experiments presented in the previous subsections further reinforce this conclusion. As progressively more model-defined parallel-process semantics are incorporated into the predictive process traces through the Parallel Process Semantics Dictionary, both prediction architectures exhibit gradual and consistent improvements across all evaluation metrics. This observation confirms that the semantic contribution of individual PSD entries is complementary and accumulative, providing empirical evidence that the Process Semantics Dictionary functions as an extensible semantic knowledge base whose progressive enrichment systematically improves predictive process representations.

Overall, the experimental results validate the central hypothesis of this study: enhancing the semantic quality of process traces is as important as improving the prediction architecture itself. By eliminating directed phantom edges and replacing arbitrary execution orders among sibling activities with their corresponding Parallel Process Semantics Dictionary entries, the proposed MSA-PPM framework enables deep learning models to learn genuine causal process semantics rather than execution-order artifacts. After all, the proposed framework constitutes an architecture-independent semantic enhancement paradigm that can be seamlessly integrated with existing as well as future deep learning architectures for predictive process monitoring.

4.3 Discussion and Implications

The experimental results demonstrate that predictive performance depends not only on the learning architecture but also on the semantic quality of its input traces. Conventional PPM preserves the observed event order as a sequential representation, even when parallel sibling activities may occur in arbitrary orders. Such representations can

introduce directed phantom edges that encode execution-order artifacts as apparent dependencies. By identifying these relationships from model-defined parallel semantics and eliminating them before predictive learning, MSA-PPM provides semantically revised traces that better reflect the underlying process structure.

This effect is consistently observed for both LSTM- and Transformer-based models. The LSTM model achieves an absolute accuracy improvement of 5.63 percentage points, while the Transformer model improves by 4.90 percentage points; improvements are also observed in the corresponding Precision, Recall, and F1-score measures. Because the predictive architectures themselves remain unchanged, these gains provide evidence that the improvement originates from the semantic preprocessing of the predictive inputs rather than from increased model complexity.

The results can be interpreted as a reduction of semantic noise in the input space. For recurrent learning, eliminating artificial sequential dependencies prevents hidden-state transitions from unnecessarily representing arbitrary orderings among parallel activities. For attention-based learning, it reduces the need to model contextual relationships arising solely from incidental parallel interleavings. In both cases, model-defined semantics allow the predictive architecture to concentrate more effectively on behaviorally meaningful dependencies.

The cumulative PSD experiments provide further evidence for this interpretation. Each parallel-semantics entry represents a discovered sibling-activity set and contributes an additional unit of model-defined knowledge to trace preprocessing. Although individual additions do not always produce monotonic improvements, the overall performance trend increases as more parallel-semantic entries are incorporated. This suggests that the predictive benefit is associated with the accumulated correction of execution-order artifacts rather than with a single particular process construct.

These findings also clarify the methodological role of the *Process Semantics Dictionary (PSD)*. The PSD is not merely a lookup table for phantom-edge elimination; it provides an explicit interface between process discovery and predictive learning. Process discovery identifies behavioral structures, the PSD represents their semantics in machine-processable form, and MSA-PTP operationalizes this knowledge by transforming predictive traces. The resulting separation between semantic interpretation and predictive learning enables the same semantic preprocessing mechanism to be applied without redesigning the downstream predictive architecture.

The present experiments validate this principle specifically for *parallel semantics* and *next-activity prediction*. They should therefore be interpreted as an operational validation of one component of the broader MSA-PPM paradigm rather than as empirical validation of all PSD semantic categories. Nevertheless, the modular formulation provides a basis for future implementations of exclusive-choice, loop, synchronization, temporal, resource, organizational, and constraint semantics, each requiring its own semantic extraction and trace-transformation mechanism.

A further implication is the stronger integration of process mining and predictive monitoring. In MSA-PPM, a discovered process model is not used solely for visualization or retrospective analysis; its semantics become operational knowledge for constructing predictive data. The resulting pipeline establishes a direct connection

Process Discovery → *PSD* → *Semantic Trace Transformation* → *Predictive Learning*, thereby transferring model-defined behavioral knowledge into data-driven prediction.

Overall, the experiments support three conclusions. First, correcting semantically inconsistent trace relationships can improve predictive performance without increasing neural-model complexity. Second, the PSD provides an explicit mechanism for transferring model-defined semantics from process discovery to predictive preprocessing. Third, the consistent improvements across LSTM and Transformer models indicate that the demonstrated benefit is not restricted to a single learning architecture. These findings support the central premise of MSA-PPM: improving the semantic validity of predictive traces can complement advances in predictive architectures as a distinct means of improving predictive process monitoring.

5 Conclusion

This paper proposed *Model-Semantics-Aware Predictive Process Monitoring (MSA-PPM)*, a paradigm that addresses the semantic gap between system-enacted event traces and model-defined process behavior. Its computational framework, Model-Semantics-Aware Process Trace Preprocessing (MSA-PTP), uses a *Process Semantics Dictionary (PSD)* to transfer semantic knowledge extracted from process models into predictive data preparation. Rather than modifying the predictive architecture, MSA-PPM transforms conventional traces into model-semantics-aware traces before predictive learning.

As an operational realization, this study implemented parallel semantics using an SICN-oriented process model discovered by the ρ-Algorithm. Model-defined sibling relationships were represented in the PSD and used to eliminate directed phantom edges caused by arbitrary execution orders among parallel activities. Next-activity prediction experiments showed absolute accuracy improvements of 5.63 and 4.90 percentage points for LSTM- and Transformer-based models, respectively, together with improvements in Precision, Recall, and F1-score. Since the predictive architectures were unchanged, these results demonstrate that improving the semantic validity of predictive traces can itself contribute to predictive performance.

The present study validates MSA-PPM specifically for parallel semantics and next-activity prediction. Future work will extend the PSD and its semantic transformation mechanisms to exclusive-choice, loop, synchronization, temporal, resource, organizational, and constraint semantics and evaluate them across additional PPM tasks and process datasets. These extensions will further investigate MSA-PPM as a reusable semantic layer connecting process discovery with predictive learning.

References

[1] Nguyen, T.-H., Pham, D.-L., Kim, K.P.: Next process-activity prediction using switch-transformer: approach, visualization, and performance evaluation. Knowledge and Information Systems **67**, 11827–11854 (2025) https://doi.org/10.1007/s10115-025-02587-z

[2] Sun, X., Yang, S., Yuke Ying, D.Y.: Next activity prediction of ongoing business processes based on deep learning. Expert Systems **41**(5), 13421–13439 (2023) https://doi.org/10.1111/exsy.13421

[3] Navarin, N., Vincenzi, B., Polato, M., Sperduti, A.: Lstm networks for data-aware remaining time prediction of business process instances. 2017 IEEE Symposium Series on Computational Intelligence (SSCI), 1–7 (2017)

[4] Bukhsh, Z.A., Saeed, A., Dijkman, R.M.: ProcessTransformer: Predictive Business Process Monitoring with Transformer Network (2021) arXiv:2104.00721

[5] Rama-Maneiro, E., Vidal, J.C., Lama, M.: Deep learning for predictive business process monitoring: Review and benchmark. IEEE Transactions on Services Computing **16**(1), 739–756 (2023) https://doi.org/10.1109/TSC.2021.3139807

[6] Márquez-Chamorro, A.E., Resinas, M., Ruiz-Cortés, A.: Predictive monitoring of business processes: A survey. IEEE Transactions on Services Computing **11**(6), 962–977 (2018) https://doi.org/10.1109/TSC.2017.2772256

[7] Impedovo, D., *et al.*: Next activity prediction: An application of shallow learning techniques against deep learning over the bpi challenge 2020. IEEE Access **11**, 117947–117953 (2023)

[8] Weytjens, H., De Weerdt, J.: Process Outcome Prediction: CNN vs. LSTM (with Attention). Lecture Notes in Business Information Processing **397**, 321–333 (2020) https://doi.org/10.1007/978-3-030-66498-5_24 arXiv:2104.06934

[9] Wang, J., Dongjin Yu*, Chengfei Liu, Xiaoxiao Sun: Predicting Outcomes of Business Process Executions Based on LSTM Neural Networks and Attention Mechanism (2021)

[10] Kim, K.P.: A structural anomaly-trace detection approach and its implications in process mining and reengineering. IEEE Access **12**, 171099–171111 (2024) https://doi.org/10.1109/ACCESS.2024.3501392

[11] Nguyen, T.-H., Kim, K.-S., Kim, K.P.: A switch-transformer predictive process monitoring model for next activity prediction in business process management. In: Proceedings of the 2024 8th International Conference on Advances in Artificial Intelligence. ICAAI '24, pp. 196–200. Association for Computing Machinery, New York, NY, USA (2025). https://doi.org/10.1145/3704137.3704186 . https://doi.org/10.1145/3704137.3704186

[12] Tax, N., Verenich, I., La Rosa, M., Dumas, M.: Predictive business process monitoring with lstm neural networks. In: CAiSE, pp. 477–492. Springer, ??? (2017)

[13] Federici, M., et al.: Transformer-based deep learning models for predictive process

monitoring. Information Systems (2023)

[14] Aversano, L., *et al.*: A data-aware explainable deep learning approach for next activity prediction. Engineering Applications of Artificial Intelligence **126**, 106758 (2023)

[15] Oved, A., Shlomov, S., Zeltyn, S., Mashkif, N., Yaeli, A.: SNAP: Semantic Stories for Next Activity Prediction (2024). https://arxiv.org/abs/2401.15621

[16] Evermann, J., Rehse, J.-R., Fettke, P.: Predicting process behaviour using deep learning. Decision Support Systems **100**, 129–140 (2017)

[17] Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Transactions on Neural Networks **20**(1), 61–80 (2009) https://doi.org/10.1109/TNN.2008.2005605

[18] Aalst, W.M.P.: Process mining: Data science in action. Springer (2016)

[19] Weijters, A.J.M.M., Ribeiro, J.T.S.: Process mining with the heuristics miner algorithm. BETA Working Paper (2006)

[20] Kalenkova, A.A., Aalst, W.M., Lomazova, I.A., Rubin, V.A.: Process mining using bpmn: relating event logs and process models. Software & Systems Modeling **16**(4), 1019–1048 (2017)

[21] Kim, K.-S.: Proportional process pattern discovery and its experimental analyses. Phd dissertation, Kyunghee University, Yongin, South Korea (February 2019). https://www.riss.kr/link?id=T15064340

[22] Kim, K.-S., Pham, D.-L., Kim, K.P.: ρ-algorithm: A sicn-oriented process mining framework. IEEE Access **9**, 139852–139875 (2021) https://doi.org/10.1109/ACCESS.2021.3119011

[23] Nguyen, T.-H., Kim, K.P.: Discovering structural formations of multi-level and compound control-flow gateways from process logs. IEEE Access **11**, 128826–128842 (2023) https://doi.org/10.1109/ACCESS.2023.3330140

[24] De Santis, F., Park, G., Aalst, W.M.P., Zanichelli, F.: Compliance-aware predictive process monitoring: A neuro-symbolic approach. In: Advanced Information Systems Engineering: 38th International Conference, CAiSE 2026, Verona, Italy, June 8–12, 2026, Proceedings, Part II, pp. 223–239. Springer, Berlin, Heidelberg (2026). https://doi.org/10.1007/978-3-032-28117-3_13 . https://doi.org/10.1007/978-3-032-28117-3_13

[25] Pan, D., Chen, Y., Li, Y., Ma, Y.: A unified representation learning framework for structure-aware predictive business process monitoring via knowledge graph-enhanced multi-task learning. Future Internet **18**(5) (2026) https://doi.org/10.

3390/fi18050263

[26] Rebmann, A., Schmidt, F.D., Glavaš, G., Aa, H.: On the potential of large language models to solve semantics-aware process mining tasks. Process Science **2**(10), 1–29 (2025) https://doi.org/10.1007/s44311-025-00019-3

[27] Kim K-.S., t.: Experimental verification and validation of the sicn-oriented process mining algorithm and system. Journal of King Saud University - Computer and Information Sciences **34**(10), 9793–9813 (2022)

[28] 4TU, C.f.R.D.: BPI Challenges. BPM (2012, 2013, 2014, 2015, 2016, 2017, 2018)

[29] Hochreiter, S., Schmidhuber, J.: Long Short-Term Memory. Neural Computation **9**(8), 1735–1780 (1997) https://doi.org/10.1162/neco.1997.9.8.1735

[30] Bukhsh, Z.A., Saeed, A., Dijkman, R.M.: Processtransformer: Predictive business process monitoring with transformer network. arXiv preprint arXiv:2104.00721 (2021)

[31] Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., Dean, J.: Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer (2017)

[32] Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers), pp. 4171–4186. Association for Computational Linguistics, ??? (2019). https://doi.org/10.18653/v1/n19-1423