Unleashing Generative and Predictive AI in Process Mining

Marco Comuzzi
Department of Industrial Engineering
Ulsan National Institute of Science and Technology (UNIST), Korea
mcomuzzi@unist.ac.kr

3rd Process Mining Summer School
August 27-28, 2026
Pohang (Korea)

Asia-Pacific Process Intelligence Network
APPI Network

Plan for Today

Predictive v. Generative AI

Predictive AI and Process Mining

Generative AI and Process Mining

Towards Agentic AI for Event Log Data Quality Assessment

Marco Comuzzi(✉), Suryeon Ra, Dariga Narmanova, Sangwoo Cho, Yeongik Hong, and Selim Jung

Ulsan National Institute of Science and Technology, Ulsan, Korea
{mcomuzzi,suryeon24,dariga,csw0617,poby7791,smileselims}@unist.ac.kr

---

RESEARCH Open Access

Explainable predictive process monitoring: a user evaluation

Williams Rizzi[1†], Marco Comuzzi[2*†], Chiara Di Francescomarino[3†], Chiara Ghidini[4†], Suhwan Lee[5†], Fabrizio Maria Maggi[4†] and Alexander Nolte[6,7†]

Abstract

Explainability is motivated by the lack of transparency of black-box machine learning approaches, which do not foster trust and acceptance of machine learning algorithms. This also happens in the predictive process monitoring field, where predictions, obtained by applying machine learning techniques, need to be explained to users, so as to gain their trust and acceptance. In this work, we carry on a user evaluation on explanation approaches for predictive process monitoring aiming at investigating whether and how the explanations provided (i) are understandable; (ii) are useful in decision making tasks; (iii) can be further improved for process analysts with different predictive process monitoring expertise levels. The results of the user evaluation show that, although explanation plots are overall understandable and useful for decision making tasks for business process management users — with and without experience in predictive process monitoring — differences exist in the comprehension and usage of different plots, as well as in the way users with different predictive process monitoring expertise understand and use them.

Keywords: Predictive process monitoring, Process mining, Explainable artificial intelligence, Explanation plots, Qualitative observational study

Introduction

AI4BPM 2024

Keynote

PPM, Quo Vadis?
Challenges and Future Directions of the Predictive Process Monitoring Research Field

Even if not a full-fledged "winter", we are surely witnessing a relative stagnation of the PPM research field, with research results mainly incremental and heavy borrowing problems and solutions from the broader machine learning field. In this talk, I will analyse the research challenges and future directions of the PPM research field, based on a work recently published, in which a set of challenges and directions have been identified and validated by research experts. After having defined the possible future of the PPM field, I will also argue for a discrepancy between where the field of PPM appears to be going and where it should probably go to play a more valuable role in the industrial process mining landscape.

Slides of the keynote talk available here.

Dr. Marco Comuzzi

Marco Comuzzi is Associate Professor and Director of the Blockchain Research Center at the Department of Industrial Engineering, Ulsan National Institute of Science and Technology (UNIST), in Ulsan (South Korea). He has also held academic full-time positions at the Eindhoven University of Technology, The Netherlands, and City, University of London, United Kingdom. His research is in the broad area of information systems design and data science, with a specific focus on process mining and blockchain technology.

Ceravolo et al. Process Science (2024) 1:2
https://doi.org/10.1007/s44311-024-00002-4 Process Science

RESEARCH Open Access

Predictive process monitoring: concepts, challenges, and future research directions

Paolo Ceravolo[1†], Marco Comuzzi[2*†], Jochen De Weerdt[3†], Chiara Di Francescomarino[4†] and Fabrizio Maria Maggi[5†]

Abstract

Predictive Process Monitoring (PPM) extends classical process mining techniques by providing predictive models that can be applied at runtime during the execution of a business process, for example, to predict the sequence of the next event(s) in a case, its outcomes, or performance-related aspects such as the remaining processing time. These predictive models go beyond process mining's inherent descriptive nature that is offered by typical process discovery, conformance checking, and model enhancement techniques. The growing interest in PPM is driven by its additional value proposition, i.e., delivering real-time information regarding the future execution of business process instances, thus allowing for better informed operational decision making and performance analysis. The rapid growth of PPM during the last ten years has left the field lacking a cohesive taxonomy and an explicit recognition of the prevailing challenges. This paper aims at closing this gap, by comprehensively defining PPM using a unified approach to the key terms and by discussing challenges and opportunities in the field. Specifically, we propose three overarching research challenges and nine research directions, which have been validated through a survey with PPM researchers.

Keywords: Predictive process monitoring, Process mining, Predictive analytics, Data science pipelines

Introduction

Predictive Process Monitoring (PPM) is an emerging subfield of Process Mining

---

Journal of Intelligent Information Systems
https://doi.org/10.1007/s10844-024-00903-7

RESEARCH

Explaining the impact of design choices on model quality in predictive process monitoring

Sungkyu Kim[1] · Marco Comuzzi[1] · Chiara Di Francescomarino[2]

Received: 19 March 2024 / Revised: 13 September 2024 / Accepted: 14 October 2024

On the Impact of Low-Quality Activity Labels in Predictive Process Monitoring*

Marco Comuzzi[1], Sungkyu Kim[1], Jonghyeon Ko[2], Musa Salamov[3], Cinzia Cappiello[3], and Barbara Pernici[3]

[1] Ulsan National Institute of Science and Technology, Ulsan, Korea
[2] Jeonju University, Jeonju, Korea
[3] Politecnico di Milano, Milan, Italy
{mcomuzzi,kimkangf3}@unist.ac.kr, whd1gus2@jj.ac.kr
{musa.salamov,cinzia.cappiello,barbara.pernici}@polimi.it

Predictive v. Generative AI

Predictive AI

Analyze existing data to make predictions about future events or outcomes

Relies on established ML techniques

Output is a prediction (recommendation, probability score, etc.)

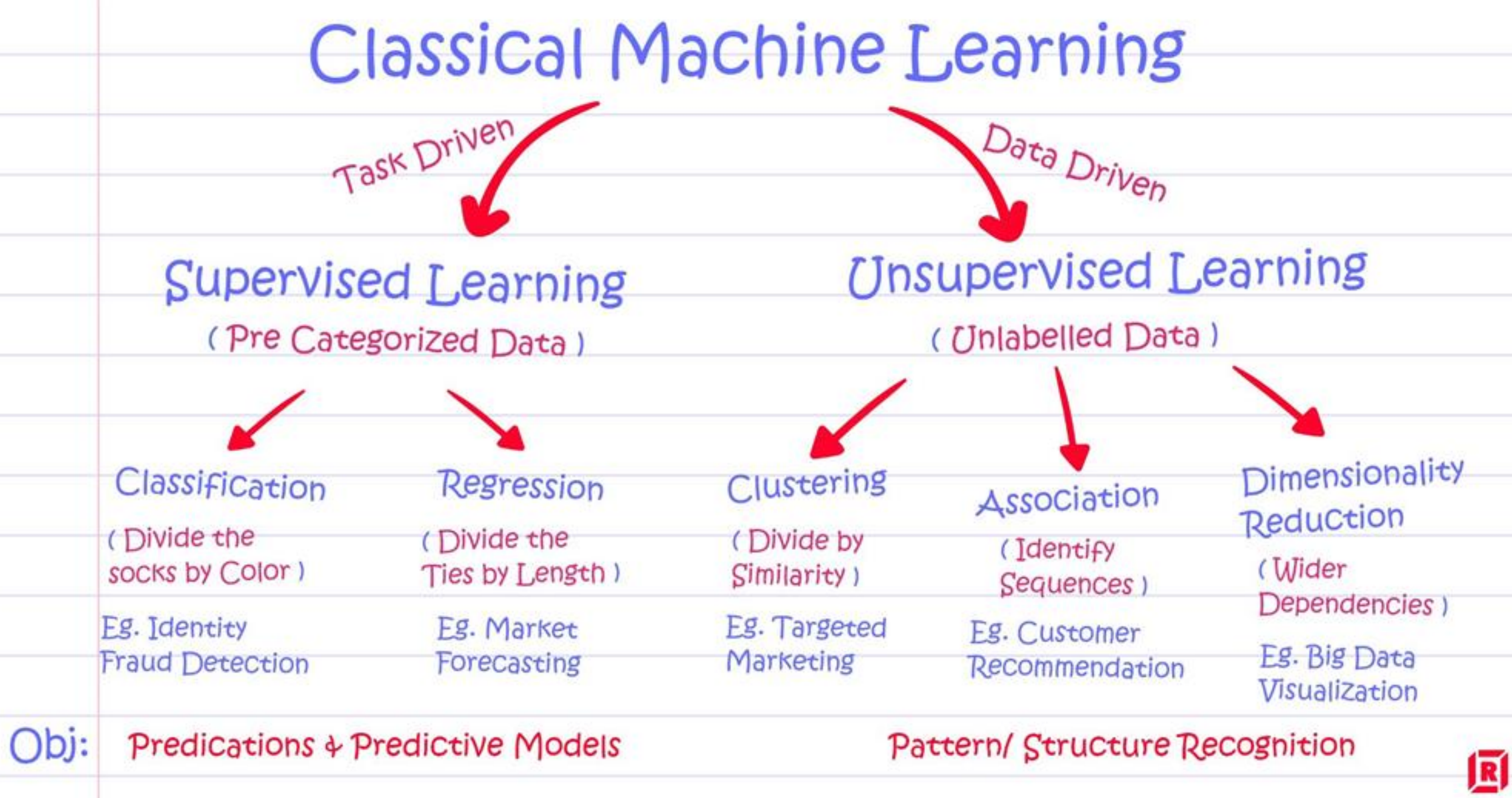
Generative AI

Aims at creating new content or data that are similar but yet distinct from training data (text, images, music, …)

Do not focus on one specific phenomenon, but exploit general knowledge learnt from huge amount of data

Relies on novel techniques (LLMs)

ML Micro-101 (Classification)


Classical Machine Learning
Task Driven
Data Driven
Supervised Learning
(Pre Categorized Data)
Unsupervised Learning
(Unlabelled Data)
Classification
(Divide the socks by Color)
Eg. Identity Fraud Detection
Regression
(Divide the Ties by Length)
Eg. Market Forecasting
Clustering
(Divide by Similarity)
Eg. Targeted Marketing
Association
(Identify Sequences)
Eg. Customer Recommendation
Dimensionality Reduction
(Wider Dependencies)
Eg. Big Data Visualization
Obj:
Predications & Predictive Models
Pattern/ Structure Recognition

Machine Learning - Classification

The problem of identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category membership is known (supervised learning)

spam v. no-spam

normal v. anomalous

Machine learning - classification

Observations are defined by a set of features (explanatory variables, predictors)

features (age, occupation, etc.)

label (salary)

39	State-gov	77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States		<=50K
50	Self-emp-not-inc	83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States		<=50K
38	Private	215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States		<=50K
53	Private	234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States		<=50K
28	Private	338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba		<=50K
37	Private	284582	Masters	14	Married-civ-spouse	Exec-managerial	Wife	White	Female	0	0	40	United-States		<=50K
49	Private	160187	9th	5	Married-spouse-absent	Other-service	Not-in-family	Black	Female	0	0	16	Jamaica		<=50K
52	Self-emp-not-inc	209642	HS-grad	9	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	45	United-States		>50K
31	Private	45781	Masters	14	Never-married	Prof-specialty	Not-in-family	White	Female	14084	0	50	United-States		>50K
42	Private	159449	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	5178	0	40	United-States		>50K
37	Private	280464	Some-college	10	Married-civ-spouse	Exec-managerial	Husband	Black	Male	0	0	80	United-States		>50K
30	State-gov	141297	Bachelors	13	Married-civ-spouse	Prof-specialty	Husband	Asian-Pac-Islander	Male	0	0	40	India		>50K
23	Private	122272	Bachelors	13	Never-married	Adm-clerical	Own-child	White	Female	0	0	30	United-States		<=50K
32	Private	205019	Assoc-acdm	12	Never-married	Sales	Not-in-family	Black	Male	0	0	50	United-States		<=50K
40	Private	121772	Assoc-voc	11	Married-civ-spouse	Craft-repair	Husband	Asian-Pac-Islander	Male	0	0	40	?		>50K
34	Private	245487	7th-8th	4	Married-civ-spouse	Transport-moving	Husband	Amer-Indian-Eskimo	Male	0	0	45	Mexico		<=50K
25	Self-emp-not-inc	176756	HS-grad	9	Never-married	Farming-fishing	Own-child	White	Male	0	0	35	United-States		<=50K
32	Private	186824	HS-grad	9	Never-married	Machine-op-inspct	Unmarried	White	Male	0	0	40	United-States		<=50K
38	Private	28887	11th	7	Married-civ-spouse	Sales	Husband	White	Male	0	0	50	United-States		<=50K
43	Self-emp-not-inc	292175	Masters	14	Divorced	Exec-managerial	Unmarried	White	Female	0	0	45	United-States		>50K
40	Private	193524	Doctorate	16	Married-civ-spouse	Prof-specialty	Husband	White	Male	0	0	60	United-States		>50K
54	Private	302146	HS-grad	9	Separated	Other-service	Unmarried	Black	Female	0	0	20	United-States		<=50K
35	Federal-gov	76845	9th	5	Married-civ-spouse	Farming-fishing	Husband	Black	Male	0	0	40	United-States		<=50K
43	Private	117037	11th	7	Married-civ-spouse	Transport-moving	Husband	White	Male	0	2042	40	United-States		<=50K
59	Private	109015	HS-grad	9	Divorced	Tech-support	Unmarried	White	Female	0	0	40	United-States		<=50K

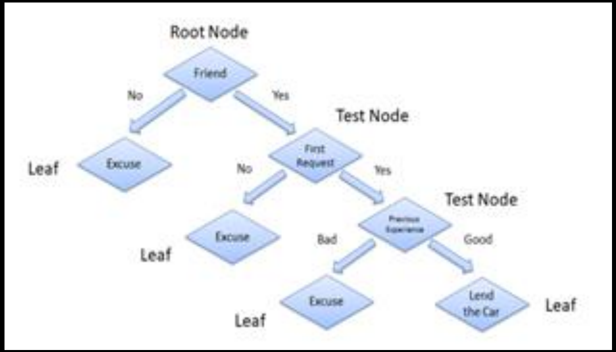
Classification - methods

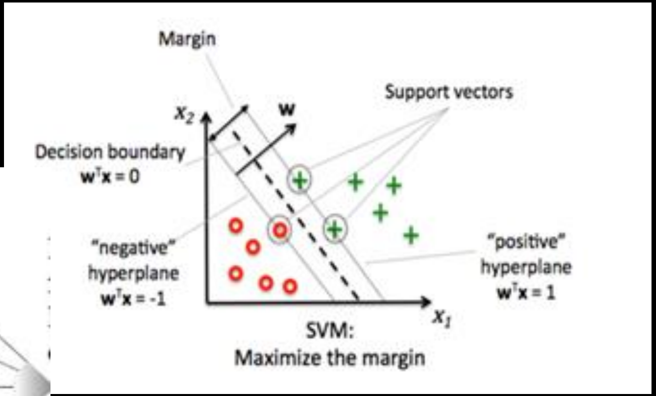
decision tree

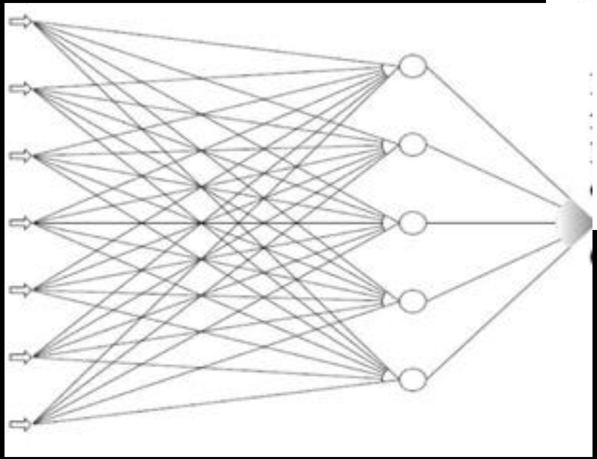
support vector machine

neural networks

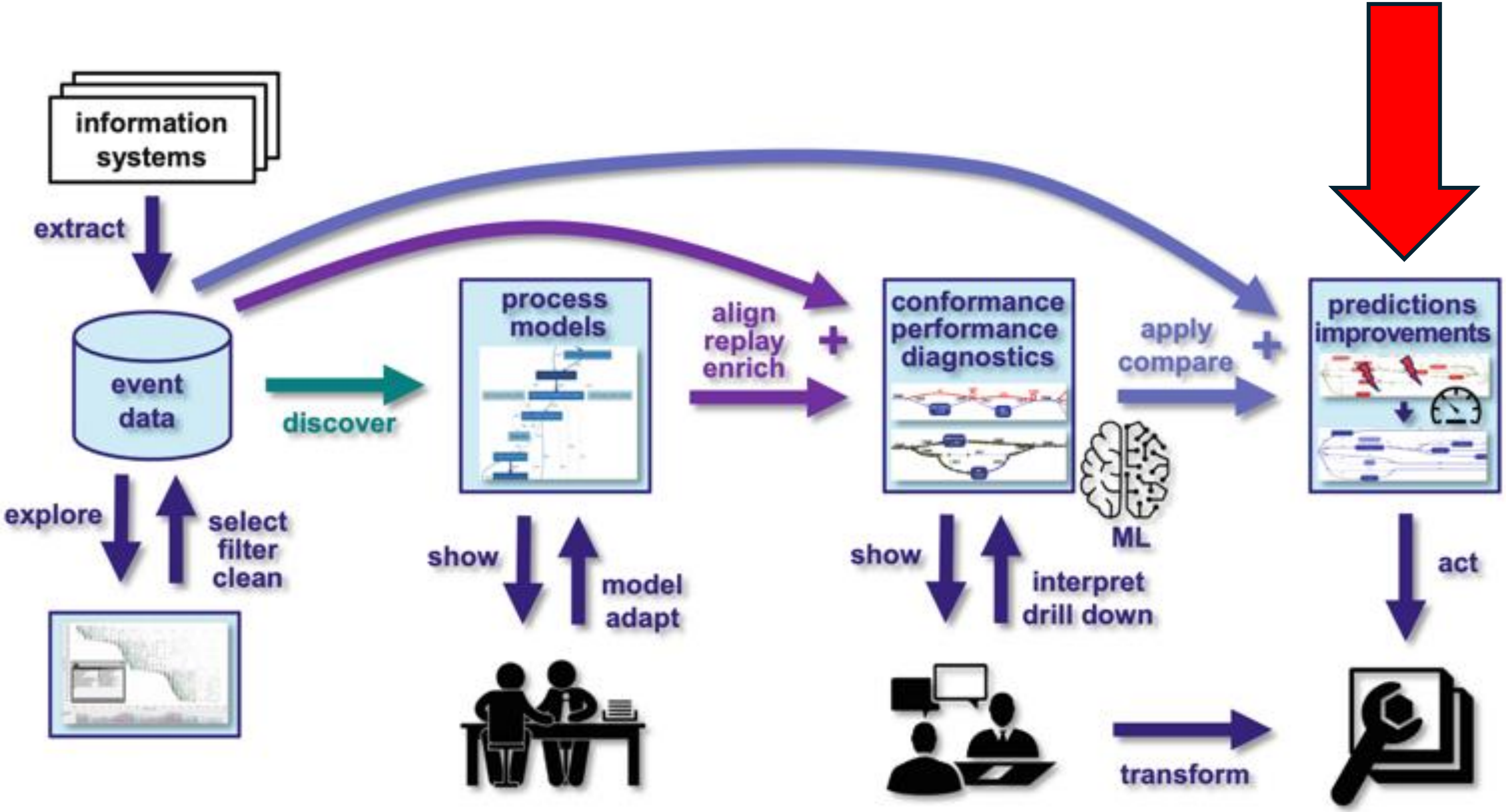
…







Predictive AI and Process Mining



Reference: van der Aalst and Carmona 2022

Predictive Process Monitoring (PPM)

What is PPM?

Event log is a dataset

Can be used to train machine learning predictive models…

...which can then be used to predict "the future" of (running) process cases

case id	event id	properties				
		timestamp	activity	resource	cost	...
1	35654423	30-12-2010:11.02	register request	Pete	50	...
	35654424	31-12-2010:10.06	examine thoroughly	Sue	400	...
	35654425	05-01-2011:15.12	check ticket	Mike	100	...
	35654426	06-01-2011:11.18	decide	Sara	200	...
	35654427	07-01-2011:14.24	reject request	Pete	200	...
2	35654483	30-12-2010:11.32	register request	Mike	50	...
	35654485	30-12-2010:12.12	check ticket	Mike	100	...
	35654487	30-12-2010:14.16	examine casually	Pete	400	...
	35654488	05-01-2011:11.22	decide	Sara	200	...
	35654489	08-01-2011:12.05	pay compensation	Ellen	200	...
3	35654521	30-12-2010:14.32	register request	Pete	50	...
	35654522	30-12-2010:15.06	examine casually	Mike	400	...
	35654524	30-12-2010:16.34	check ticket	Ellen	100	...
	35654525	06-01-2011:09.18	decide	Sara	200	...
	35654526	06-01-2011:12.18	reinitiate request	Sara	200	...
	35654527	06-01-2011:13.06	examine thoroughly	Sean	400	...
	35654530	08-01-2011:11.43	check ticket	Pete	100	...
	35654531	09-01-2011:09.55	decide	Sara	200	...
	35654533	15-01-2011:10.45	pay compensation	Ellen	200	...
4	35654641	06-01-2011:15.02	register request	Pete	50	...
	35654643	07-01-2011:12.06	check ticket	Mike	100	...
	35654644	08-01-2011:14.43	examine thoroughly	Sean	400	...
	35654645	09-01-2011:12.02	decide	Sara	200	...
	35654647	12-01-2011:15.44	reject request	Ellen	200	...
...	...	...	...	...	...	...

What can We Predict About a Business Process?

"Which will be the next activity that will be executed?"

Predicting activities that will be executed

"When will the next activity be executed? What will be the timestamp of the last activity of this case?"

Predicting timestamps

"What would be the outcome of this case, e.g., would it be possible to discharge this patient by tomorrow?"

Predict process outcomes

Why PPM?

"What is the next activity that will be executed?"

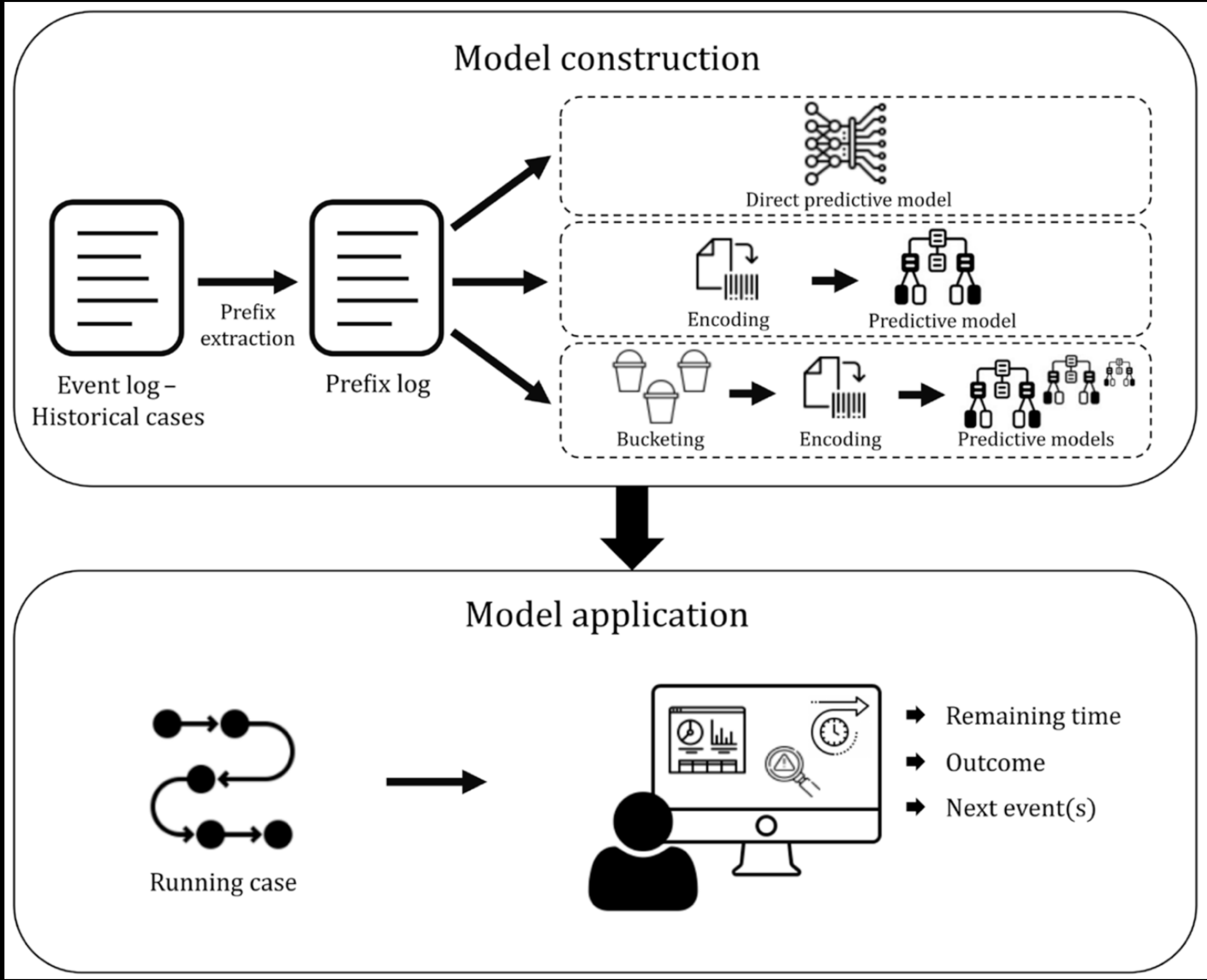
"Get ready" if the next predicted is "critical"

"When will the next activity be executed?"

Early warning to clients when we predict their order will be delivered late

"Would it be possible to discharge this patient by tomorrow?"

Monitor satisfaction of Service Level Objectives



Ceravolo, P., Comuzzi, M., De Weerdt, J., Di Francescomarino, C., & Maggi, F. M. (2024). Predictive process monitoring: concepts, challenges, and future research directions. Process Science, 1(1), 2.

Case-Level v. Event-Level Attributes

Case-Level

Attributes that have the same value in all events belonging to the same case

Event-Level

Attributes that may have a different value in every event

Case	Case attributes		Event attributes			
ID	D.O.B.	Sex	Activity	Completion Time	Resource	Cost
1	20/02/1982	M	Registration	1/1/2017 9:13:00	John	15
1	20/02/1982	M	Antibiotics	1/1/2017 9:14:20	Mark	25
1	20/02/1982	M	Triage	1/1/2017 9:16:00	Mary	10
1	20/02/1982	M	Release	1/1/2017 9:18:05	Kate	20
1	20/02/1982	M	Return	1/1/2017 9:18:50	John	20
1	20/02/1982	M	Blood test	1/1/2017 9:19:00	Kate	15
2	03/08/1967	F	Registration	2/1/2017 16:55:00	John	25
2	03/08/1967	F	Triage	2/1/2017 17:00:00	Mary	25
2	03/08/1967	F	Antibiotics	3/1/2017 9:00:00	Mark	10
2	03/08/1967	F	Release	3/1/2017 9:01:50	Kate	15

Trace Prefixes

From a trace in an event log of length n-events, we can derive up to n-1 **prefixes**

A prefix of length $k < n$ is the sub-trace from the 1^{st} to k-th event

Prefixes are important to maximize the number of observations available to train a PPM model that can be generated from an event log

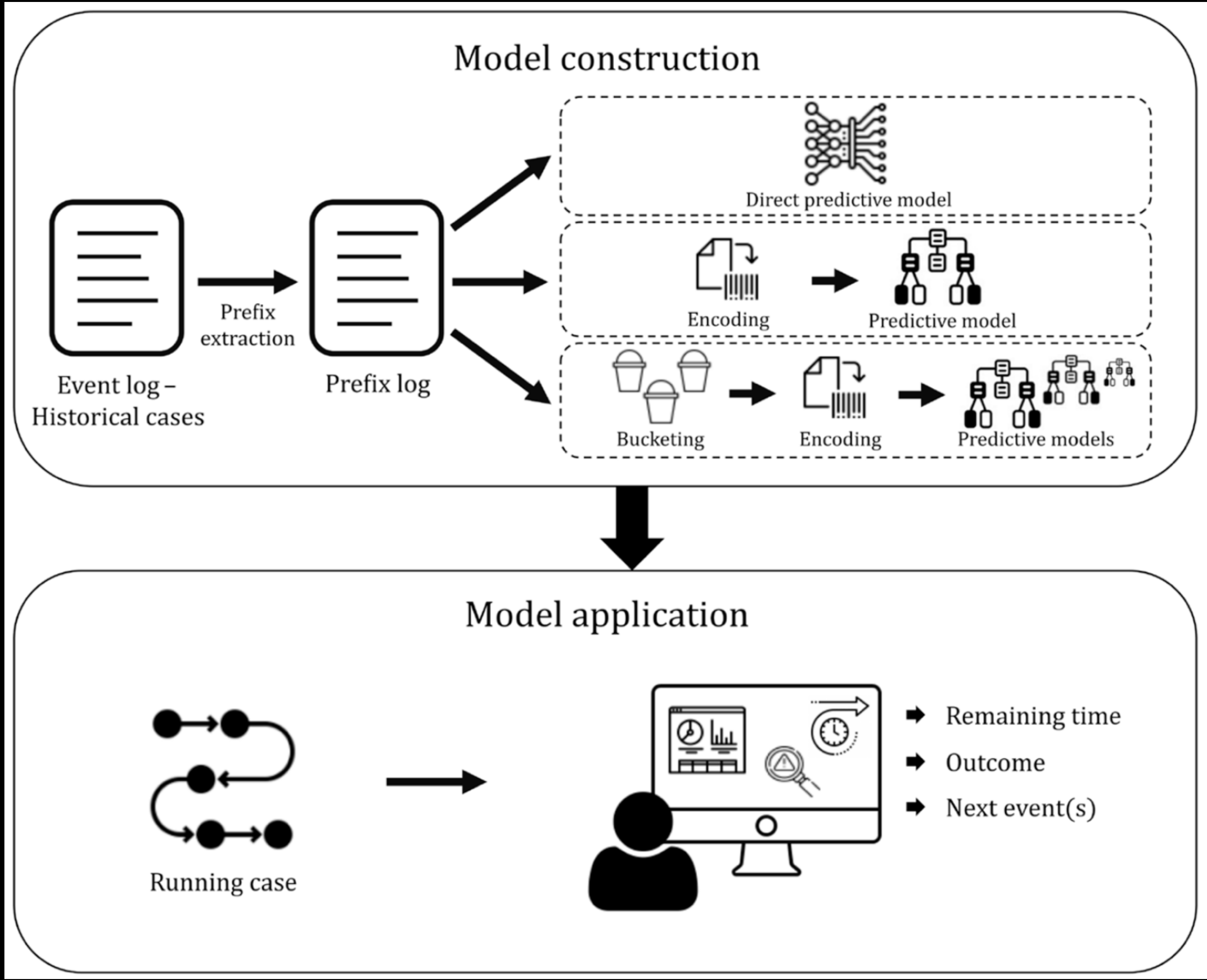
	Activity, timestamp, resource,			
Trace (n=4)	A, t=1, Alice	B, t=4, Alice	C, t=6, Bob	B, t=7, Chris
Prefix (k=3)	A, t=1, Alice	B, t=4, Alice	C, t=6, Bob	
Prefix (k=2)	A, t=1, Alice	B, t=4, Alice		
Prefix (k=1)	A, t=1, Alice			

Temporal Split

Events in an event log are timestamped

To simulate a real-life situation, the events used for training a PPM model should always precede the ones used for testing the model

Case	Case attributes		Event attributes			
ID	D.O.B.	Sex	Activity	Completion Time	Resource	Cost
1	20/02/1982	M	Registration	1/1/2017 9:13:00	John	15
1	20/02/1982	M	Antibiotics	1/1/2017 9:14:20	Mark	25
1	20/02/1982	M	Triage	1/1/2017 9:16:00	Mary	10
1	20/02/1982	M	Release	1/1/2017 9:18:05	Kate	20
1	20/02/1982	M	Return	1/1/2017 9:18:50	John	20
1	20/02/1982	M	Blood test	1/1/2017 9:19:00	Kate	15
2	03/08/1967	F	Registration	2/1/2017 16:55:00	John	25
2	03/08/1967	F	Triage	2/1/2017 17:00:00	Mary	25
2	03/08/1967	F	Antibiotics	3/1/2017 9:00:00	Mark	10
2	03/08/1967	F	Release	3/1/2017 9:01:50	Kate	15

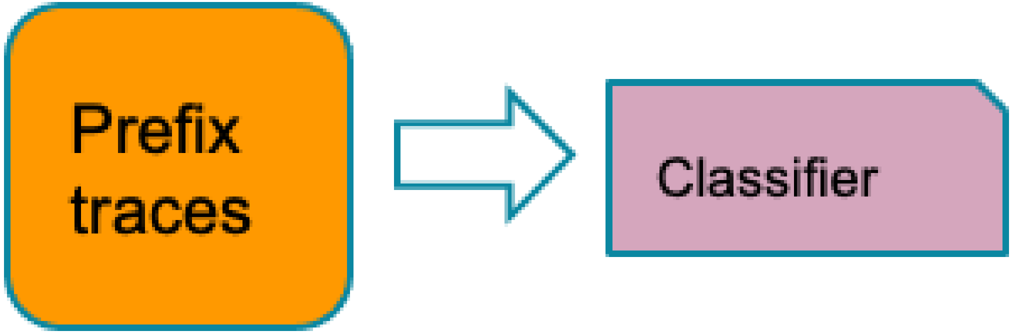


Ceravolo, P., Comuzzi, M., De Weerdt, J., Di Francescomarino, C., & Maggi, F. M. (2024). Predictive process monitoring: concepts, challenges, and future research directions. Process Science, 1(1), 2.

Bucketing of Prefixes

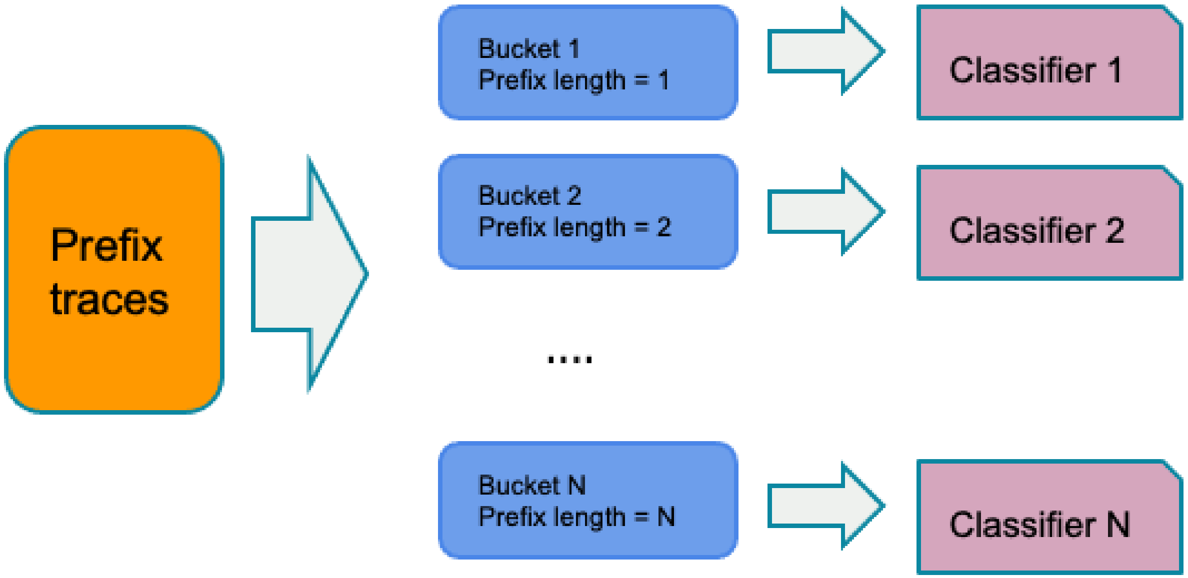
Prefix traces can be grouped into buckets to train multiple classifiers, one for each bucket

The simplest choice? Zero-Bucketing
(1 single bucket with all the prefixes)



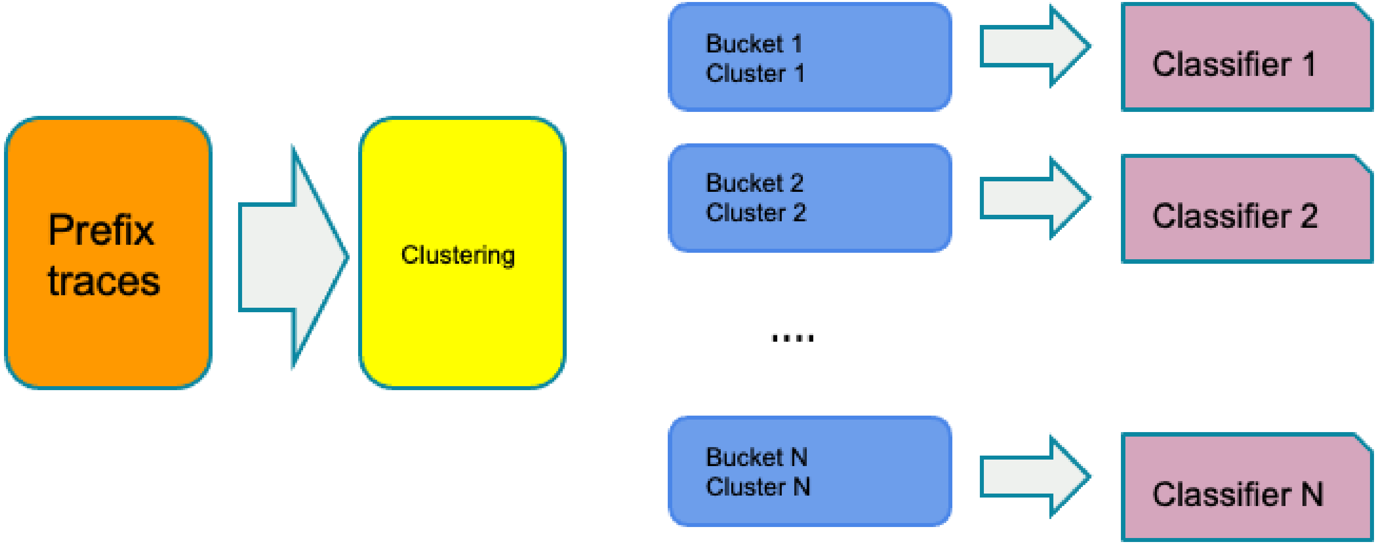
Bucketing by Prefix Length

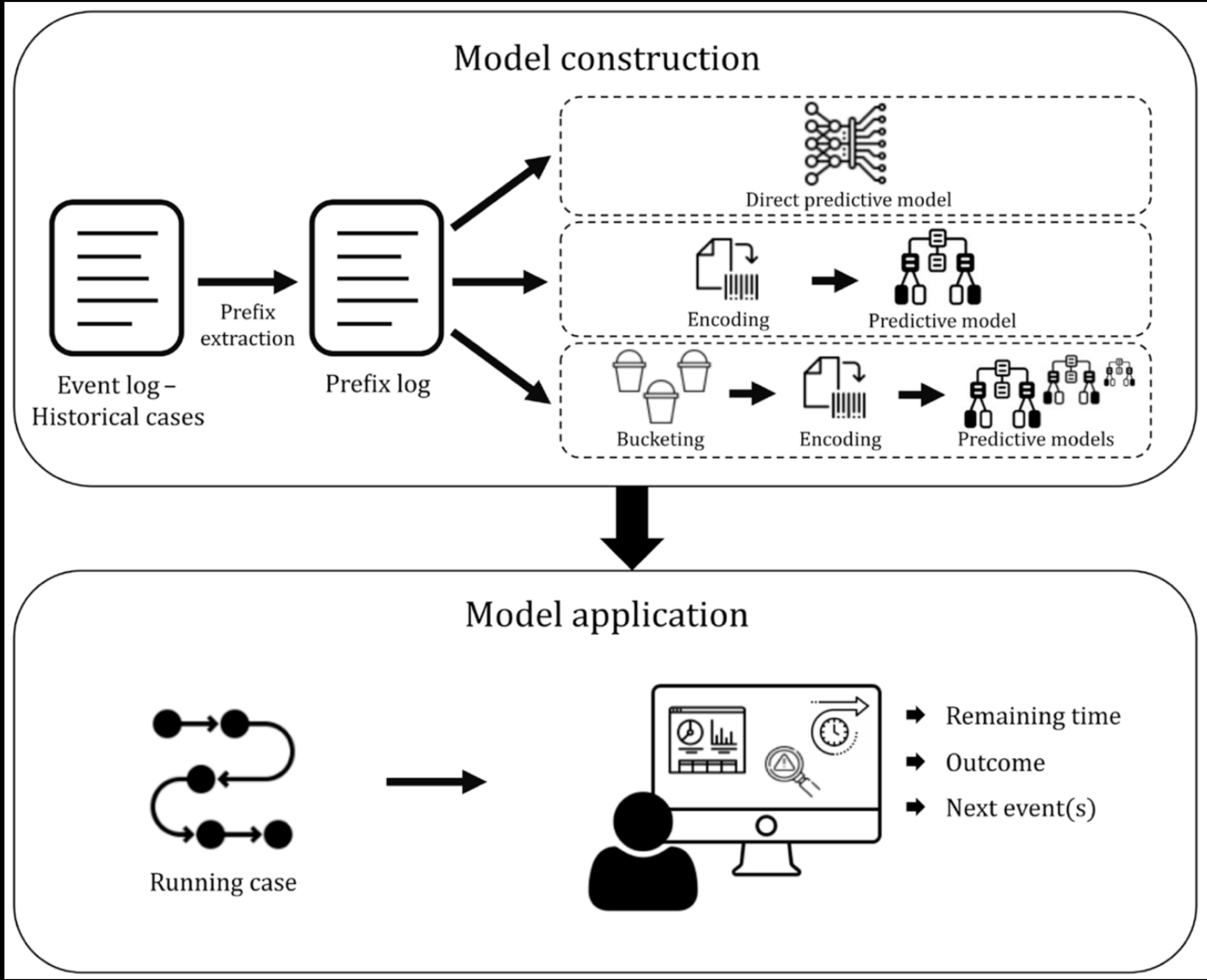
A bucket contains all the prefixes of the same length



Bucketing by Clustering

Buckets determined by applying a trace clustering method





Ceravolo, P., Comuzzi, M., De Weerdt, J., Di Francescomarino, C., & Maggi, F. M. (2024). Predictive process monitoring: concepts, challenges, and future research directions. Process Science, 1(1), 2.

Outcome prediction

Assume a (usually binary) case outcome label is available
Encoding each prefix, use outcome label as classification label

Binary classification problem (positive v. negative outcome)

Can we predict the outcome of a running case and how early during the case execution can we get a correct prediction?

Example

case id	activity	timestamp	resource	cost
1	A	1	Alice	100
2	A	3	Alice	90
1	B	4	Alice	50
1	C	6	Bob	60
1	B	7	Chris	70
2	C	8	Chris	100

case id	outcome
1	1
2	0

Trace 1: ABCB
Trace 2: AC

often “outcomes” are available as case-level attributes in an event log, or can be engineered from the data

Encoding of Prefixes (Sequence Encoding)

Sequence encoding combines the task of **feature engineering** and **numerical encoding** of traditional machine learning

First, decide which features to generate from attributes available in an event log, then encode those features numerically

The prefixes in each bucket must be encoded into a fixed-length numerical feature vector

Prefixes in a bucket may have different length

A prefix abstraction may be needed to reduce the prefixes to the same length (or zero-padding!)

Aggregation Sequence Encoding

Prefixes are abstracted by consider all the events

Attribute values are aggregated

Example

Numerical attributes: avg, sum, max, min

Categorical attributes: frequency or occurrence

	Activity, timestamp, resource, cost				Outcome
Trace 1	A, t=1, Alice, 100	B, t=4, Alice, 50	C, t=6, Bob, 60	B, t=7, Chris, 70	1
Trace 2	A, t=3, Alice, 90	C, t=8, Chris, 100	-	-	0

Activity B occurs twice in trace 1

Alice executes 2 activities in trace 1

	A_1	B_1	C_1	t_avg	Alice_1	Bob_1	Chris_1	c_max	Outcome
Trace 1	1	2	1	4.5	2	1	1	100	1
Trace 2	1	0	1	5.5	1	0	1	100	0

Time aggregated using average
Cost aggregated using max

(3 + 8) / 2 = 5.5

max(90,100) = 100

Index Sequence Encoding

Encode all the information in an event log, maintaining the order of events

If prefixes in a bucket do not have the same length, zero-padding is needed

Lossless encoding: given the feature vector, we can perfectly reconstruct all the events of a trace

	Activity, timestamp, resource, cost				Outcome
Trace 1	A, t=1, Alice, 100	B, t=4, Alice, 50	C, t=6, Bob, 60	B, t=7, Chris, 70	1
Trace 2	A, t=3, Alice, 90	C, t=8, Chris, 100	D, t=9, Dana, 45	C, t=10, Bob, 70	0

	Act_1*	Res_1*	t_1	c_1	Act_2*	Res_2*	t_2	c_2	Act_3*	Res_3*	t_3	c_3	Act_4*	Res_4*	t_4	c_4	Outcome
Trace 1	A	Alice	1	100	B	Alice	4	50	C	Bob	6	60	B	Chris	7	70	1
Trace 2	A	Alice	3	90	C	Chris	8	100	D	Dana	9	45	C	Bob	10	70	0

* Features from one-hot encoding of activity and resource omitted for readability

	Event 1 (task label, timestamp, resource label, cost)	Event 2	Event 3	Event 4	Outcome
Trace 1	A, t=1, Alice, 100	B, t=4, Alice, 50	C, t=6, Bob, 60	B, t=7, Chris, 70	1
Trace 2	A, t=3, Alice, 90	C, t=8, Chris, 100			0

	Act_1*	Res_1*	t_1	c_1	Act_2*	Res_2*	t_2	c_2	Act_3*	Res_3*	t_3	c_3	Act_4*	Res_4*	t_4	c_4	Outcome
Trace 1	A	Alice	1	100	B	Alice	4	50	C	Bob	6	60	B	Chris	7	70	1
Trace 2	A	Alice	3	90	C	Chris	8	100	0	0	0	0	0	0	0	0	0

* Features from one-hot encoding of activity and resource omitted for readability

Last-State Sequence Encoding

Prefixes are abstracted using only the last m events (m = 2 in the example below)

A simple numerical encoding is to encode numerical attributes as-is and categorical attributes using one-hot encoding

	Activity, timestamp, resource, cost				Outcome
Trace 1	A, t=1, Alice, 100	B, t=4, Alice, 50	C, t=6, Bob, 60	B, t=7, Chris, 70	1
Trace 2	A, t=3, Alice, 90	C, t=8, Chris, 100	-	-	0

	A_1	B_1	C_1	t_1	Alice_1	Bob_1	Chris_1	c_1	A_2	B_2	C_2	t_2	Alice_2	Bob_2	Chris_2	c_2	Outcome
Trace 1	0	0	1	6	0	1	0	60	0	1	0	7	0	0	1	70	1
Trace 2	1	0	0	3	1	0	0	90	0	0	1	8	0	0	1	100	0

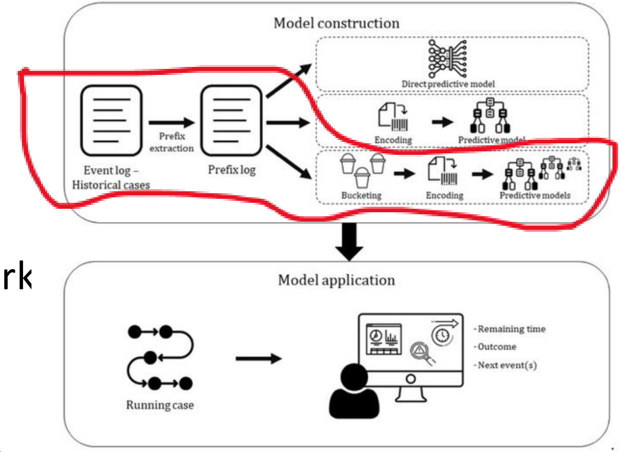
Static Attribute Encoding

For case-level attributes, which remain the same in every prefix

	Client age	Client type	Activity, timestamp, resource, cost		Outcome
Trace 1	29	Gold	A, t=1, Alice, 100	B, t=4, Alice, 50	1
Trace 2	57	Silver	A, t=3, Alice, 90	C, t=8, Chris, 100	0

	C-age	Gold	Silver	Act_1*	Res_1*	t_1	c_1	Act_2*	Res_2*	t_2	c_2	Outcome
Trace 1	29	1	0	A	Alice	1	100	B	Alice	4	50	1
Trace 2	57	0	1	A	Alice	3	90	C	Chris	8	100	0

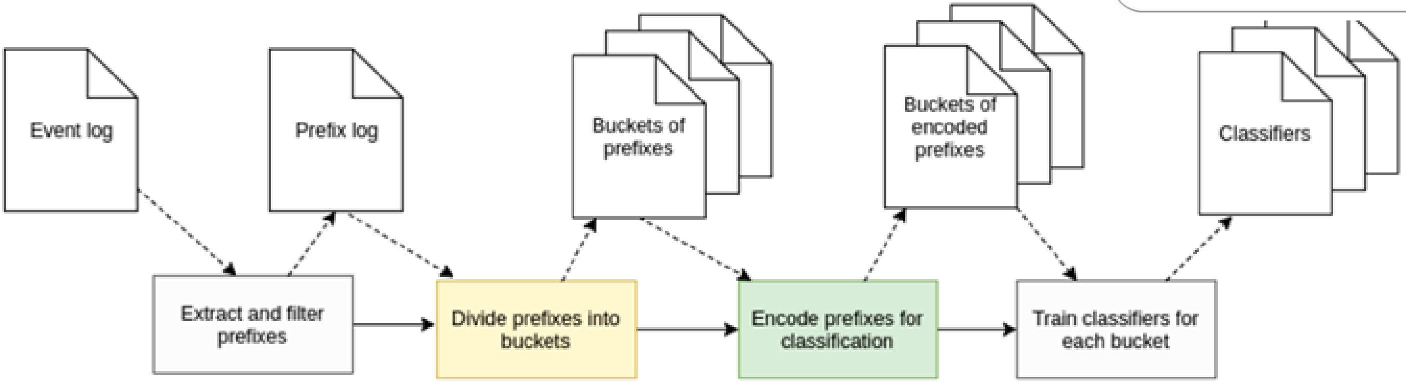
Model Construction: Training the Classifier(s)



Using the encoded prefixes, we can train/test any classifier

Logistic regression, Decision Tree, Random Forest, Neural Network

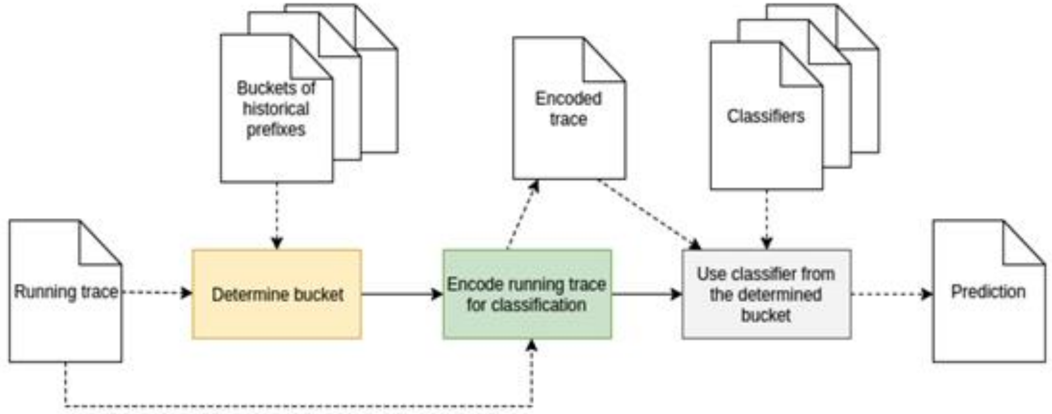
…

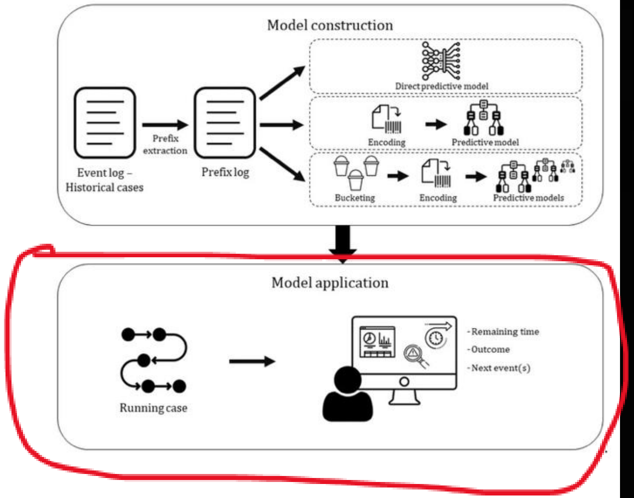


Reference: Teinemaa, I., Dumas, M., Rosa, M. L., & Maggi, F. M. (2019). Outcome-oriented predictive process monitoring: Review and benchmark. ACM Transactions on Knowledge Discovery from Data (TKDD), 13(2), 1-57.

Model Application

Extract prefixes, divide them into groups (buckets), encode prefixes into feature-label vectors, train classifiers (one per bucket)





Reference: Teinemaa, I., Dumas, M., Rosa, M. L., & Maggi, F. M. (2019). Outcome-oriented predictive process monitoring: Review and benchmark. ACM Transactions on Knowledge Discovery from Data (TKDD), 13(2), 1-57.

Evaluation Metrics

Predictions in Outcome-Oriented PPM should be:

Accurate (Accuracy, Precision, Recall, AUC)

Early and **stable** across prefixes

Break-down the classifier accuracy by prefix length

Experimental results

Publicly available event logs (Business Process Intelligence Challenge)

Different classifier tested (XGBoost, Random Forest, ...)

Outcome-Oriented Predictive Process Monitoring: Review and Benchmark

IRENE TEINEMAA and MARLON DUMAS, University of Tartu
MARCELLO LA ROSA, The University of Melbourne
FABRIZIO MARIA MAGGI, University of Tartu

Predictive business process monitoring refers to the act of making predictions about the future state of ongoing cases of a business process, based on their incomplete execution traces and logs of historical (completed) traces. Motivated by the increasingly pervasive availability of fine-grained event data about business process executions, the problem of predictive process monitoring has received substantial attention in the past years. In particular, a considerable number of methods have been put forward to address the problem of outcome-oriented predictive process monitoring, which refers to classifying each ongoing case of a process according to a given set of possible categorical outcomes—e.g., Will the customer complain or not? Will an order be delivered, canceled, or withdrawn? Unfortunately, different authors have used different datasets, experimental settings, evaluation measures, and baselines to assess their proposals, resulting in poor comparability and an unclear picture of the relative merits and applicability of different methods. To address this gap, this article presents a systematic review and taxonomy of outcome-oriented predictive process monitoring methods, and a comparative experimental evaluation of eleven representative methods using a benchmark covering 24 predictive process monitoring tasks based on nine real-life event logs.

CCS Concepts: • **Applied computing → Business process monitoring**;

Additional Key Words and Phrases: Business process, predictive monitoring, sequence classification

ACM Reference format:
Irene Teinemaa, Marlon Dumas, Marcello La Rosa, and Fabrizio Maria Maggi. 2019. Outcome-Oriented Predictive Process Monitoring: Review and Benchmark. *ACM Trans. Knowl. Discov. Data* 13, 2, Article 17 (March 2019), 57 pages.
https://doi.org/10.1145/3301300

1 INTRODUCTION

A business process is a collection of inter-related events, activities, and decision points that involve a number of actors and objects, which collectively lead to an outcome that is of value to a customer [11]. A typical example is an order-to-cash process: a process that starts when a purchase order is received and ends when the product/service is delivered and the payment is confirmed. An

This research is funded by the Australian Research Council under Grant DP150103356, the Estonian Research Council under Grant IUT20-55, and European Regional Development Fund (Dora Plus Program).
Authors' addresses: I. Teinemaa, M. Dumas, and F. M. Maggi, University of Tartu, J. Liivi 2, 50409 Tartu, Estonia; emails: {irene.teinemaa, marlon.dumas, f.m.maggi}@ut.ee; M. La Rosa, The University of Melbourne, Level 10, Doug McDonell Building, Victoria 3010, Australia; email: marcello.larosa@unimelb.edu.au.

1556-4681/2019/03-ART17 $15.00
https://doi.org/10.1145/3301300

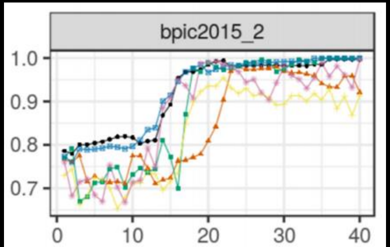
Experimental results

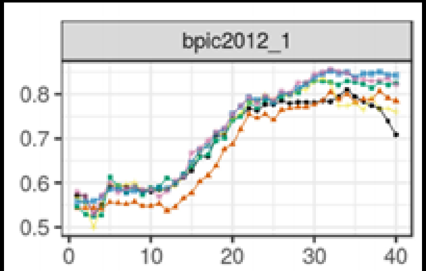
Performance varies across event logs and classifiers used

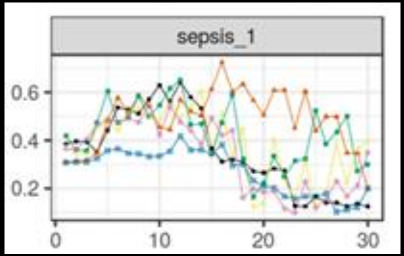
Performance higher and more predictable for low variability logs (= fewer variants)

Impossible to identify best bucketing/encoding method

performance more accurate when longer prefixes are considered, but not always







Next activity prediction

Sequence prediction, multi-class classification

Encode the N events before the one we want to predict

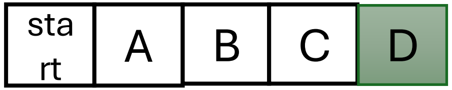
Multi-class classification problem
(>2 activity labels in a log normally)

Can we predict which would the next step in a case given the recent "history" of that case?

Next-Activity and Suffix Prediction

No specific requirements imposed on event logs,
every event log has activity labels

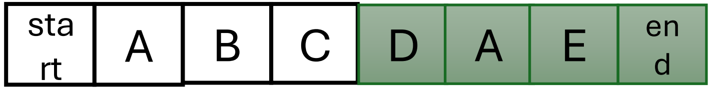
Next-Activity Prediction



Use event log data to create a model to predict which is the next activity that will be executed

Suffix Prediction

Use event log data to create a model to predict the sequence of activities that will be executed until the case termination



How to Approach Next-Activity Prediction

Simple Setting

Multi-class classification problem using window-based encoding of log traces

State-of-the-Art Setting

Multi-class classification problem solved using Recurrent Neural Networks (RNNs)

Window-Based Encoding for NAP

Similar to Last-State Encoding in outcome-oriented prediction, with m = window size

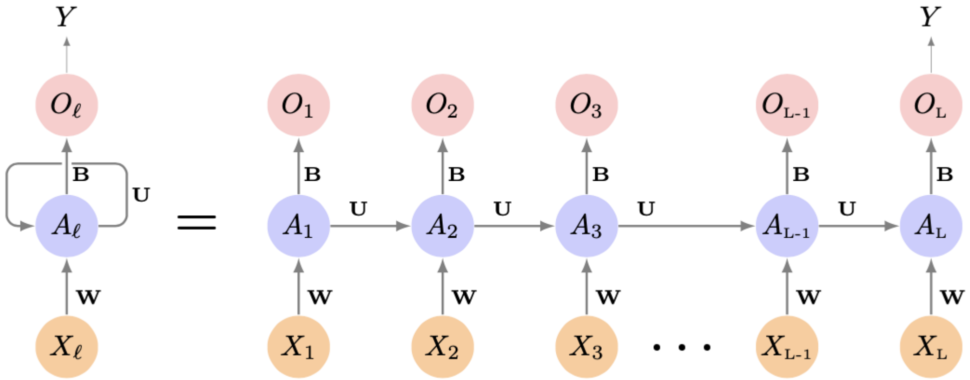
We encode m events of a prefix using a sliding window from the start to the end of a trace

case_id	event_type	timestamp
173688	activity A	10/01/2011 19:45
173688	activity B	10/01/2011 20:17
173688	activity C	10/13/2011 18:37
173691	activity A	10/01/2011 19:43
173691	activity B	10/01/2011 22:36
173691	activity C	10/10/2011 19:30
173691	activity C	10/10/2011 22:17
...	...	...

case_id	event_1	event_2	event_3	duration	next_event
173688	3	5	6	17212	3
173691	3	5	6	12947	6
173691	5	6	6	12941	6
...	...	...	...	...	...

Act_1*	Res_1*	t_1	c_1	Act_2*	Res_2*	t_2	c_2	Act_3*	Res_3*	t_3	c_3
A	Alice	1	100	B	Alice	4	50	C	Bob	6	60
A	Alice	3	90	C	Chris	8	100	D	Dana	9	45

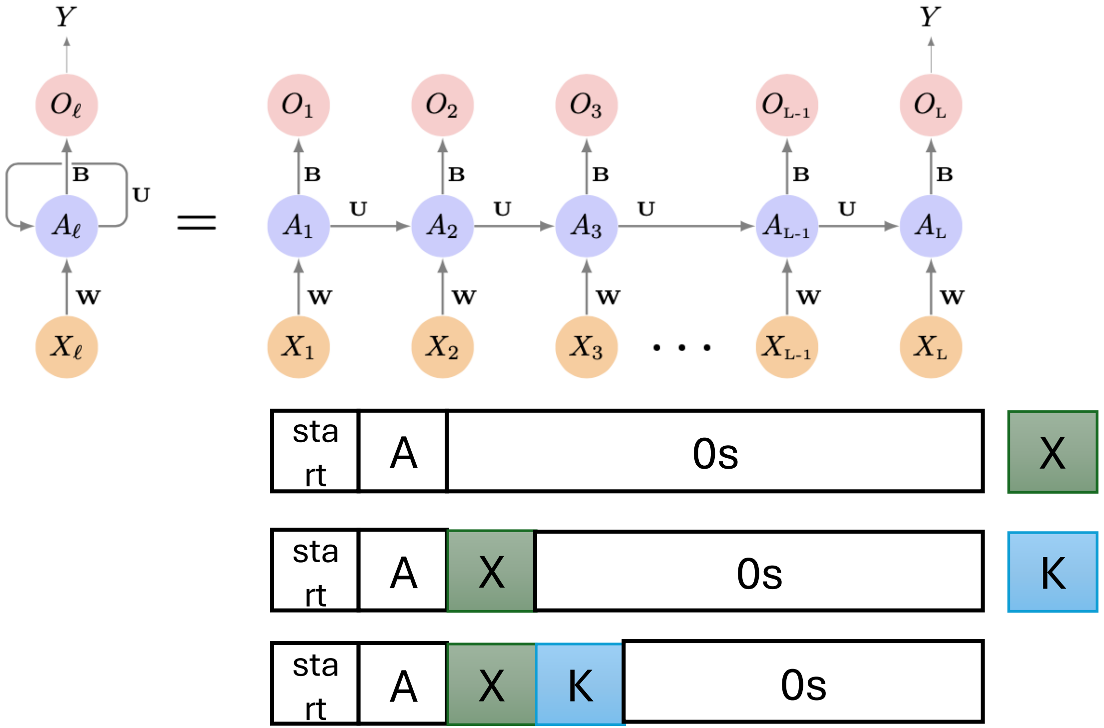
Reference: Tama et al. (2020)

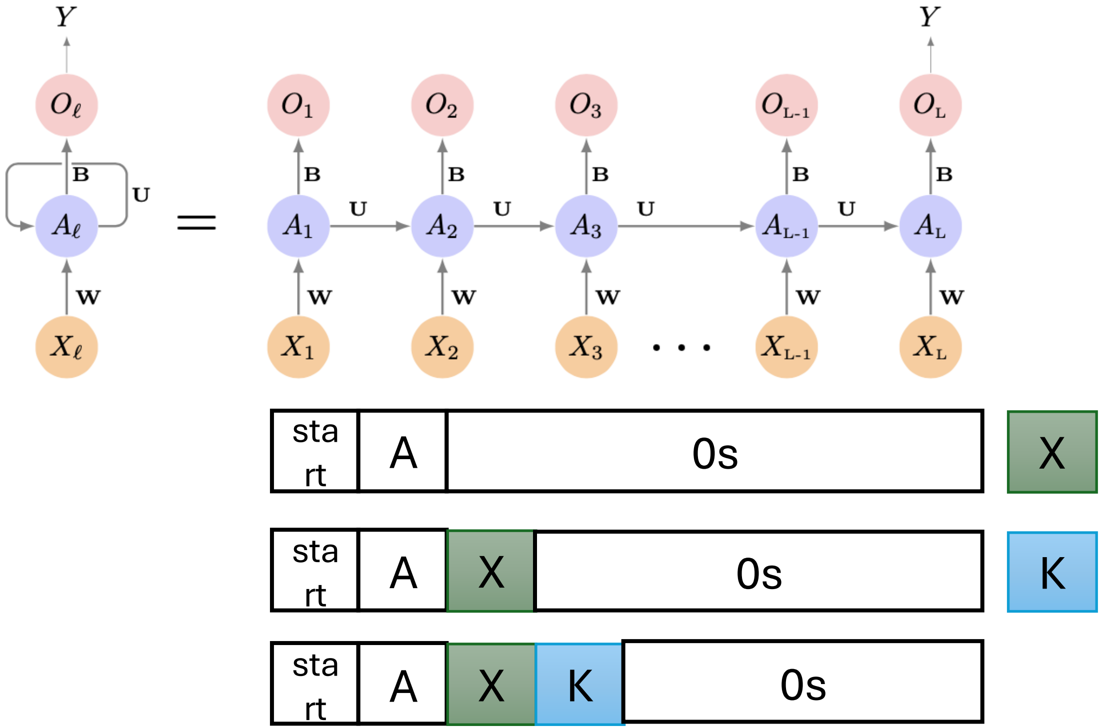


sta rt	A	0s

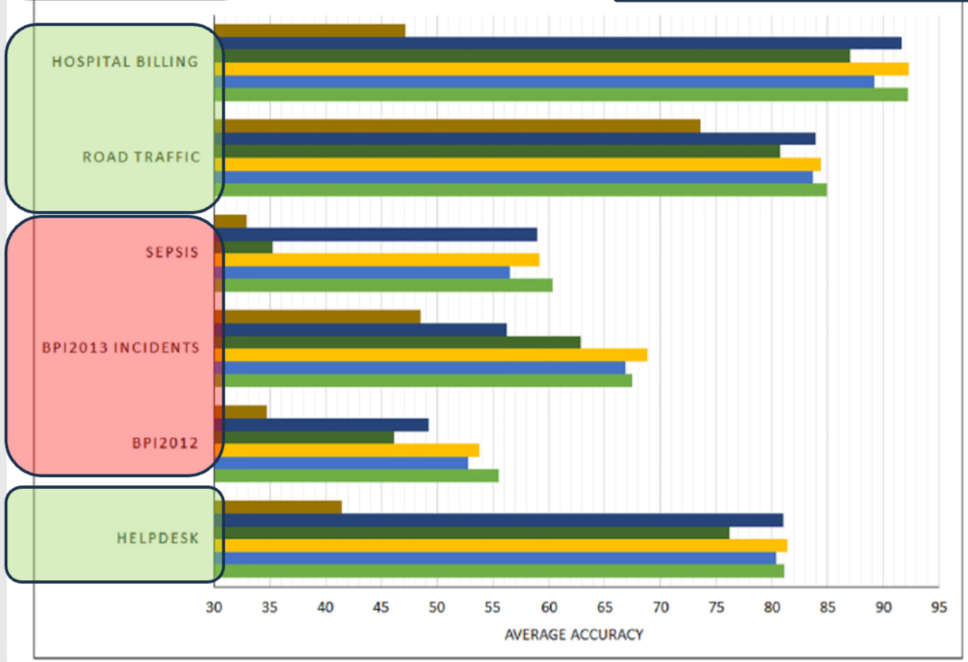
sta rt	A	B	0s

sta rt	A	B	C	0s


Y
O_ℓ
B
U
A_ℓ
W
X_ℓ
=
O_1 O_2 O_3 O_{L-1} O_L
A_1 A_2 A_3 A_{L-1} A_L
X_1 X_2 X_3 X_{L-1} X_L
sta rt
A
0s
X
K


Y
O_ℓ
B
U
A_ℓ
W
X_ℓ
=
O_1 O_2 O_3 O_{L-1} O_L
A_1 A_2 A_3 A_{L-1} A_L
X_1 X_2 X_3 … X_{L-1} X_L
start A 0s X
start A X 0s K
start A X K 0s

Event log	Event types	Trace variants	Number of variants to 80% cases	Variability (ratio)	Mean trace length	Median trace length
Helpdesk	9	154	5	Low (0.032)	3.60	3
Hospital Billing	16	288	3	Low (0.01)	5.00	5
Road Traffic	11	44	2	Low (0.045)	3.47	2
Sepsis	16	846	635	High (0.75)	14.49	13
BPIC2013	13	2278	767	High (0.33)	20.04	11
BPIC2012	36	4366	1748	High (0.40)	8.68	6



Tama, B. A., & Comuzzi, M. (2019). An empirical comparison of classification techniques for next event prediction using business process event logs. Expert Systems with Applications, 129, 233-245.

LLMs and Process Mining

Where is Generative AI?

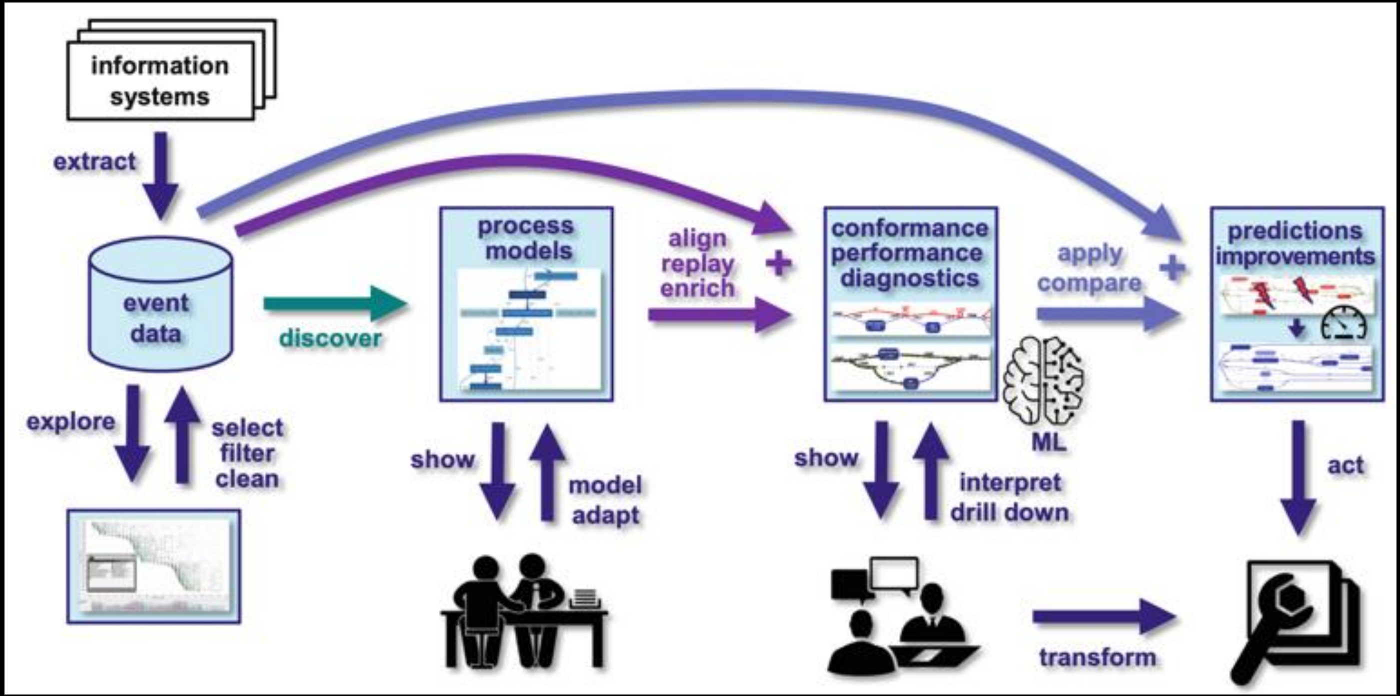
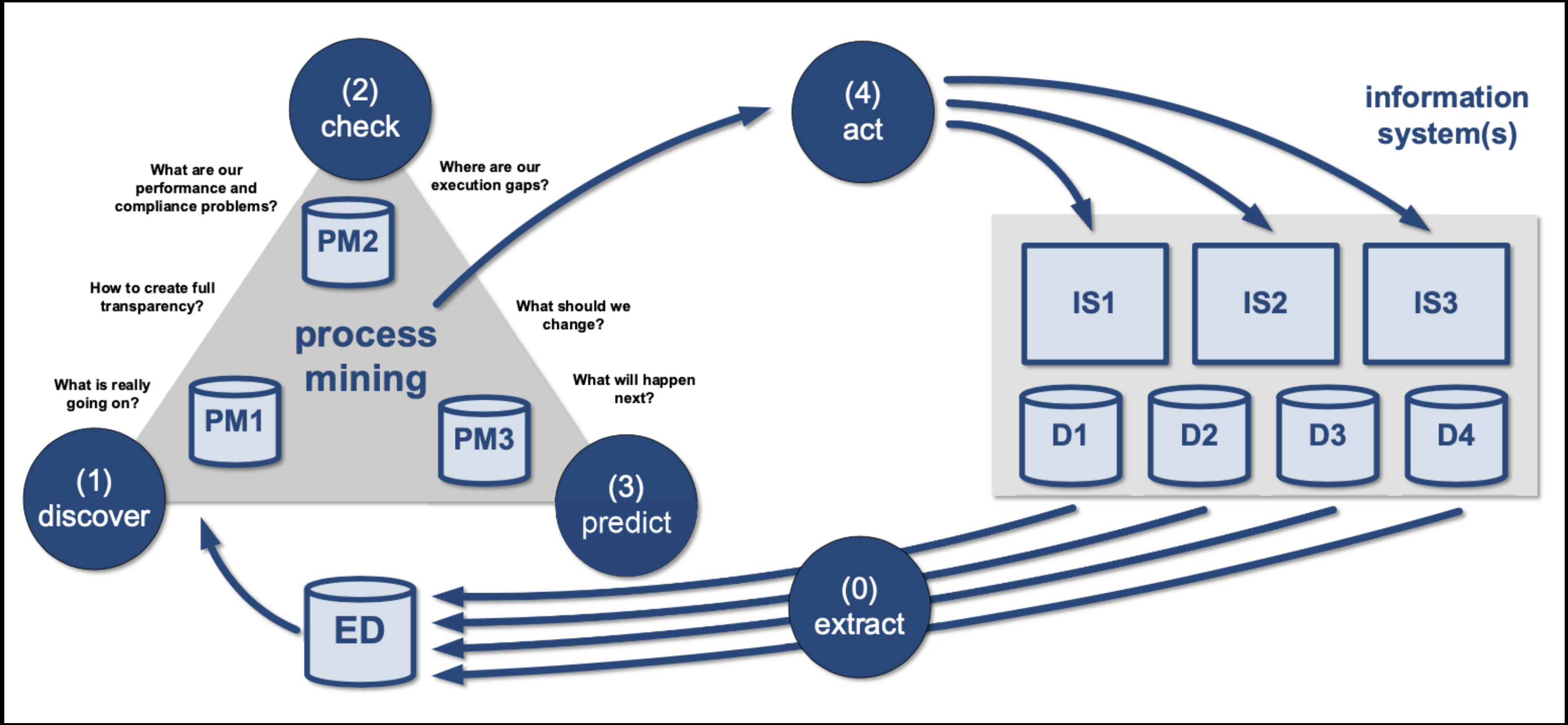
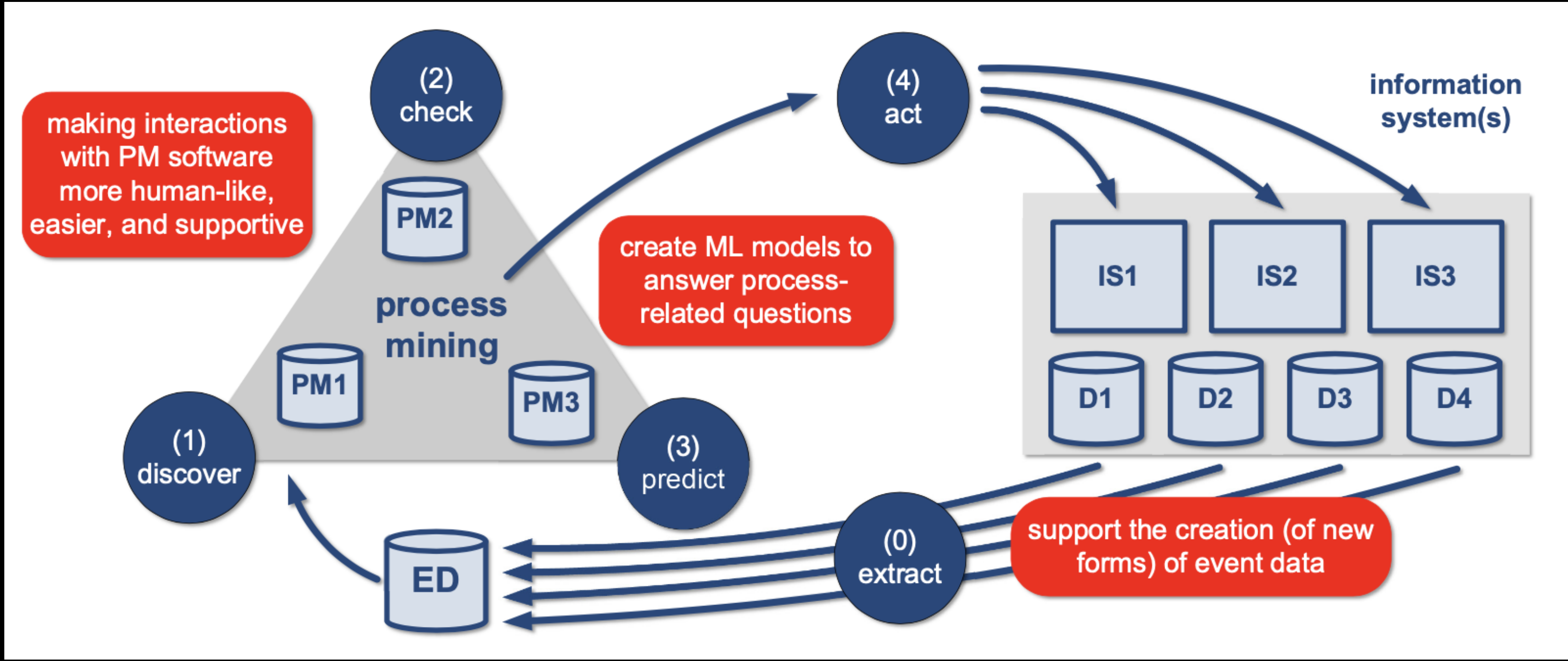


Figure from van der Aalst and Carmona Eds. (2022) Process Mining Handbook (Springer)



van der Aalst, W. (2023). Process Management after ChatGPT: How Generative and Predictive AI Relate to Process Mining. *LinkedIn Pulse Article (7-12-2023)*, 7.

LLMs in Process Mining



From: van der Aalst, W. (2023). Process Management after ChatGPT: How Generative and Predictive AI Relate to Process Mining. *LinkedIn Pulse Article (7-12-2023)*, 7.

LLMs for PM: Technical Challenges

Input/Output Limitations and Prompt Design

Output Style

Output Verification and Evaluation

Limitations of Context window size (input/output)

There is a limit on the size of the input that a LLM can handle (and the size of the output that it can produce)

LLMs may struggle with context windows > 8K words (tokens)

The longer the input and output, the more we pay!

We cannot simply give a whole event log in input and/or ask to produce an event log as output

Extract “compact” abstractions from event logs

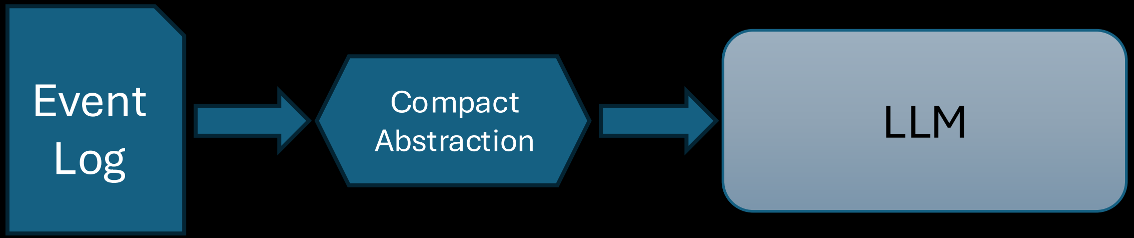
Abstractions for Prompt Generation

Directly-Follows Graphs (DFGs)

Process Variants

Process Models

(Customized when necessary)



DFGs as Text

DFG capture directly-follows relations and are annotated with frequency and performance (time-related) information (possibly aggregated)

This information can be textually represented to aid an LLM

Create Fine −> Send Fine (frequency = 103392 performance = 7568635.65)
Send Fine −> Insert Fine Notification (frequency = 79757 performance = 1501626.95)
Insert Fine Notification −> Add penalty (frequency = 72334 performance = 5184000.0)
Add penalty −> Send for Credit Collection (frequency = 57182 performance = 45566346.44)
Create Fine −> Payment (frequency = 46952 performance = 905663.45)

Trace Variants as Text

Similarly to DFGs, process variants can be annotated with frequency in the log and time-related performance information

This information can be textually represented to aid an LLM

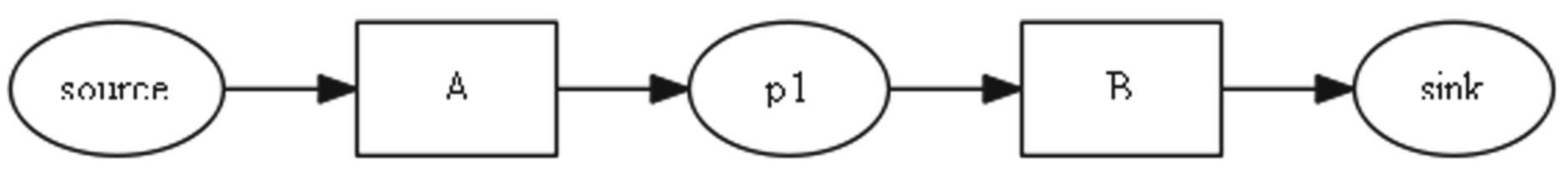
Example:

```
1000 traces: CheckOrder -> AnalyzeOrder –> Pick Items -> …
750 traces: CheckOrder -> GetMoreInfo -> CheckOrder -> …
…
```

Abstractions of Process Models

Petri nets, process trees, BPMN models etc. can be abstracted textually

Declarative process models are “textual” by design



places: [p1, sink, source]
transitions: [(A, ’A’), (B, ’B’)]
arcs: [(A, ’A’)−>p1, (B, ’B’)−>sink, p1−>(B, ’B’), source−>(A, ’A’)]
initial marking: [’source:1’]
final marking: [’sink:1’]

Event Log Abstractions in PM4Py

LLM Integration (`pm4py.llm`)

The following methods provide just the abstractions of the given objects:

- `pm4py.llm.abstract_dfg()`; provides the DFG abstraction of a traditional event log
- `pm4py.llm.abstract_variants()`; provides the variants abstraction of a traditional event log
- `pm4py.llm.abstract_log_attributes()`; provides the abstraction of the attributes/columns of the event log
- `pm4py.llm.abstract_log_features()`; provides the abstraction of the machine learning features obtained from an event log
- `pm4py.llm.abstract_case()`; provides the abstraction of a case (collection of events)
- `pm4py.llm.abstract_ocel()`; provides the abstraction of an object-centric event log (list of events and objects)
- `pm4py.llm.abstract_ocel_ocdfg()`; provides the abstraction of an object-centric event log (OC-DFG)
- `pm4py.llm.abstract_ocel_features()`; provides the abstraction of an object-centric event log (features for ML)
- `pm4py.llm.abstract_event_stream()`; provides an abstraction of the (last) events of the stream related to a traditional event log
- `pm4py.llm.abstract_temporal_profile()`; provides the abstraction of a temporal profile model
- `pm4py.llm.abstract_petri_net()`; provides the abstraction of a Petri net
- `pm4py.llm.abstract_declare()`; provides the abstraction of a DECLARE model
- `pm4py.llm.abstract_log_skeleton()`; provides the abstraction of a log skeleton model
- `pm4py.llm.explain_visualization()`; explains one of the visualizations provided in PM4Py using LVMs.

Output Style

Direct Provision of Insights

Code Generation

Automated Hypothesis Generation

Direct Provision of Insights

LLM prompt formulated combining event log abstractions and queries to provide a direct answer (insight)

`“In the context of this event log, what is the meaning of activity A?”`

`“Can you analyze the event log to detect any unusual behavior?”`

Queries refined or adjusted based on the reply

LLM’s answers may differ across sessions

Make sure LLM focuses only on the provided event log abstraction

Code Generation

LLMs are good at creating code (SQL queries, Python scripts, ...)

LLMs can be asked to create structured SQL queries to be executed on the database holding event log information

LLMs can be asked to generate Python scripts that can perform PM tasks on the input event data

Assume we have a process mining event log containing the following attributes: case id, activity, timestamp, resource. Can you write an example of this log including 2 cases and 3 events for each case?

Sure! Here's an example of a process mining event log with two cases, each containing three events:

case_id	activity	timestamp	resource
1	A	2025-03-21 10:00:00	Alice
1	B	2025-03-21 10:15:00	Bob
1	C	2025-03-21 10:30:00	Alice
2	A	2025-03-21 11:00:00	Bob
2	B	2025-03-21 11:20:00	Alice
2	C	2025-03-21 11:40:00	Bob

Now, can you write a SQL query that operates on the a database stable storing the events of the log in the example. This query produces a new column in the event log that, for each event, reports the number of times that the resource of this event has executed an activity in the same case in the past.

Sure! The following SQL query computes a new column, `resource_past_count`, which counts how many times the same resource has executed an activity in the same case **before** the current event's timestamp.

SQL Query:

sql Copy Edit

```sql
WITH PastEvents AS (
    SELECT
        e1.case_id,
        e1.activity,
        e1.timestamp,
        e1.resource,
        COUNT(e2.activity) AS resource_past_count
    FROM events e1
    LEFT JOIN events e2
        ON e1.case_id = e2.case_id  -- Same case
        AND e1.resource = e2.resource  -- Same resource
        AND e2.timestamp < e1.timestamp  -- Past events only
    GROUP BY e1.case_id, e1.activity, e1.timestamp, e1.resource
)
SELECT * FROM PastEvents
ORDER BY case_id, timestamp;
```

Automated Hypothesis Generation

Combine direct insight provision and code generation to prompt LLMs for *verifiable* autonomous hypothesis generation

1. LLM generates hypotheses (insights)

2. LLM generates the code to verify the hypotheses generated

3. (LLM can suggest refinements of the hypothesis based on the verification results)

Output Verification and Evaluation

How do we assess the quality of the output produced by the LLM?

Has the LLM hallucinated?

Hypothesis does not make sense

Is the output inaccurate?

Code is not executable or does not give the correct output if executed

Automatic Verification (for structured output)

Automated testing of the LLM-generated output

Executability of the generated code can be easily tested automatically

Formal correctness of declarative process constraints or other process insights with a formal representation may also be verified automatically

Human Evaluation

Essential direct querying and hypothesis generation

Recall: LLM’s ability to identify expected insights

Precision: Correctness of insights

Extremely expensive

Self-Evaluation and LLM as a Judge

Ask the LLM to evaluate (or reason about) their output

Chain-of-thought: ask LLM to detail their reasoning

Ensembling: using results from multiple LLMs

Self-reflection: ask LLM to review its or another’s output

Confidence scores: ask LLM to assess the insights’ reliability, discarding uncertain outputs

LLM-as-judge: use another LLM (usually known to be well-performing) at the task to judge the quality of the output

LLMs for Processing Mining Benchmark

How good are different LLMs at solving "typical" process mining tasks?

Standard prompts

LLM as a judge (for automated evaluation)

Maintain a leaderboard

https://github.com/fit-alessandro-berti/pm-llm-benchmark/tree/main

Process Mining Tasks for LLM Benchmark

1. General-purpose qualitative tasks
2. Process mining domain knowledge questions
3. Process model generation
4. Process model understanding
5. Hypotheses generation
6. Fairness assessment
7. Visual prompts

LLMs for Event Log Preparation

Background

Suriadi, S., Andrews, R., ter Hofstede, A. H., & Wynn, M. T. (2017). Event log imperfection patterns for process mining: Towards a systematic approach to cleaning event logs. Information systems, 64, 132-150.

Event log entities

		Case	Event	Rela-tion	Case attrs.	Posi-tion	Act. name	Timestamp	Resource	Event attrs.
Event log quality issues	Missing data	I1	I2	I3	I4	I5	I6	I7	I8	I9
	Incorrect data	I10	I11	I12	I13	I14	**I15**	**I16**	**I17**	I18
	Imprecise data			I19	I20	I21	**I22**	I23	I24	I25
	Irrelevant data	I26	**I27**							

Table. Manifestation of quality issues in event log entities.

Bose, R.J.C. et al. (2013). Wanna improve process mining results?. In *2013 IEEE symposium on computational intelligence and data mining (CIDM)* (pp.127-134). IEEE.

Pattern	Quality issue(s)
Form-based Event Capture	I16, I27
Inadvertent Time Travel	I6
Unanchored Event	I23, I6
Scattered Event	I2
Elusive Case	I3
Scattered Case	I12
Collateral Events	I27
Polluted Label	I15, I17
Distorted Label	I15
Synonymous Labels	I15
Homonymous Label	I22

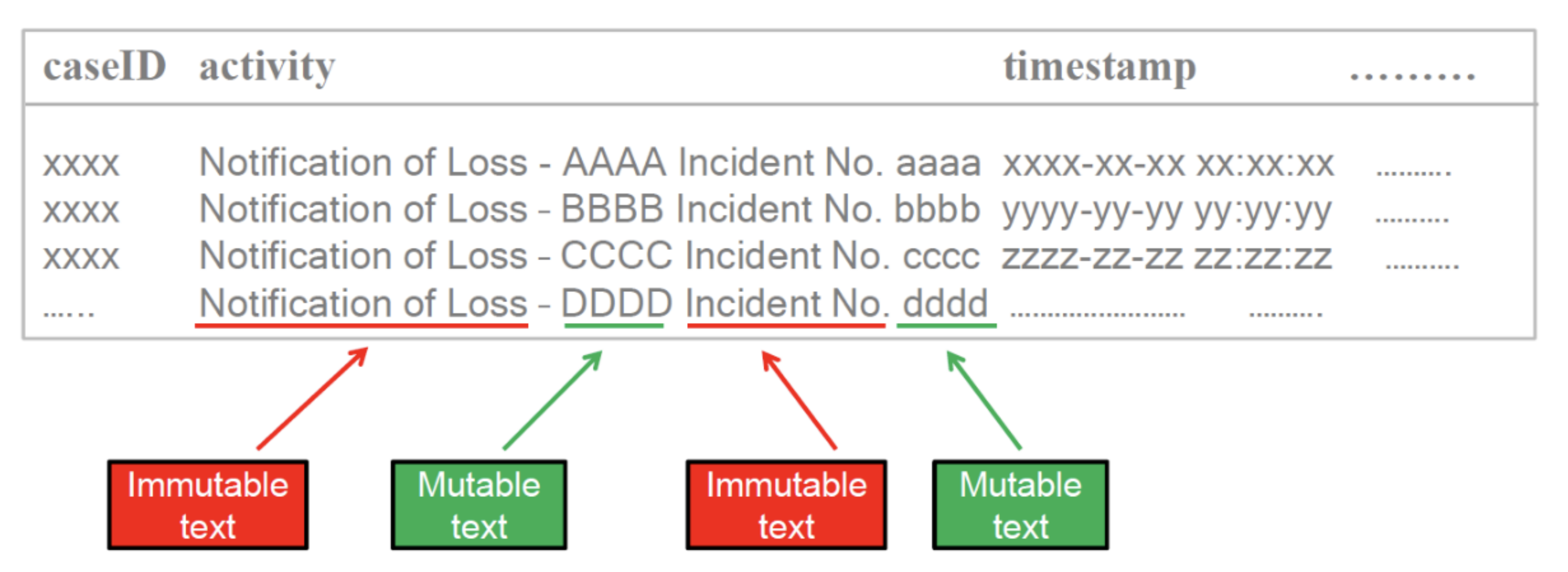
Form-Based Event Capture

Episode ID	Event	Timestamp	Description	...
ID1	Primary Survey	2012-11-23 15:42:38	....	....
ID1	Airway Clear	2012-11-23 15:42:38	....	....
ID1	...	2012-11-23 15:42:38	[illegible]	[illegible]
[illegible]	Primary Survey	2012-11-24 09:58:33	....	[illegible]
[illegible]	Airway Clear	2012-11-24 09:58:33	....	[illegible]
[illegible]	...	2012-11-24 09:58:33	....	....
[illegible]	Procedure 1	2012-11-24 09:58:33	Completed on 2012-11-24 06:58:34	....

These events are recorded on a form …

… and all have the same timestamp.

Polluted Labels



Distorted Labels

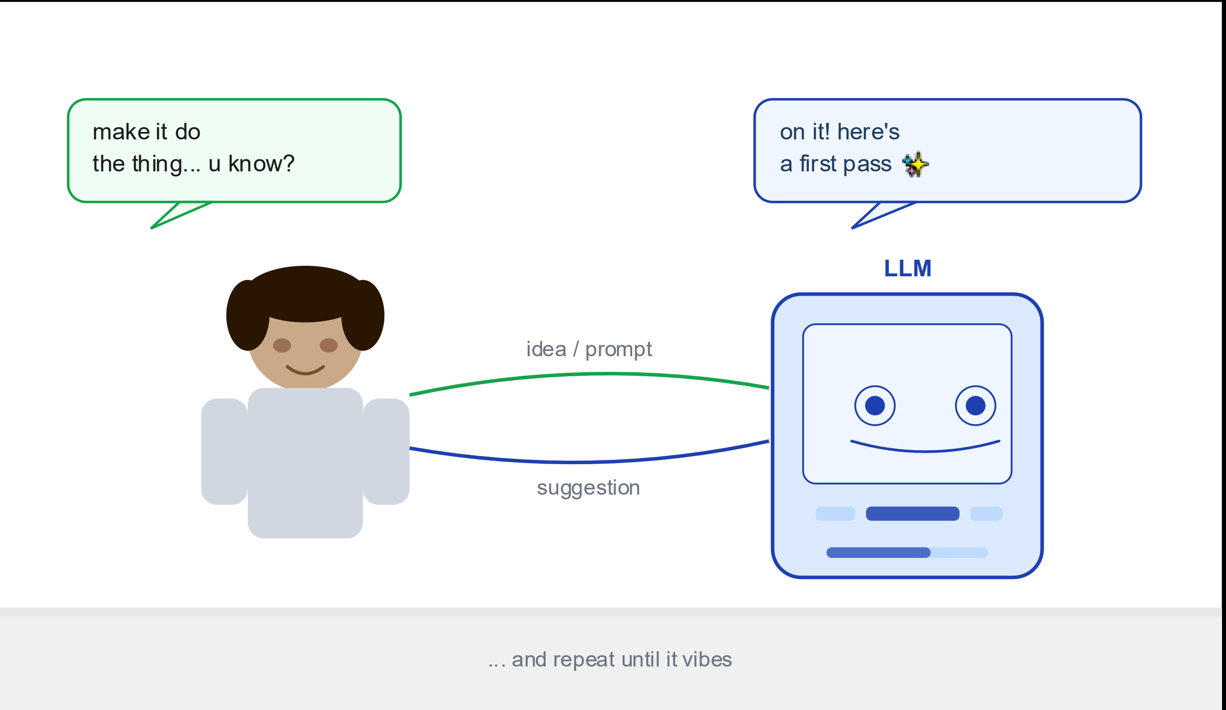
caseID	activity	timestamp	Description
1234567	a/w inv to cls.	06/09/2013 12:33:17	....
8912345	a/w inv to cls	06/09/2013 13:10:23	....
1234567	XX – Further Information Required	06/09/2013 13:15:00	....
8912345	XX – Further Infomation Required	13/09/2013 07:24:36	....

Problem Formulation

Detecting event log imperfection patterns today requires **domain experts** manually crafting detection rules for each imperfection type.

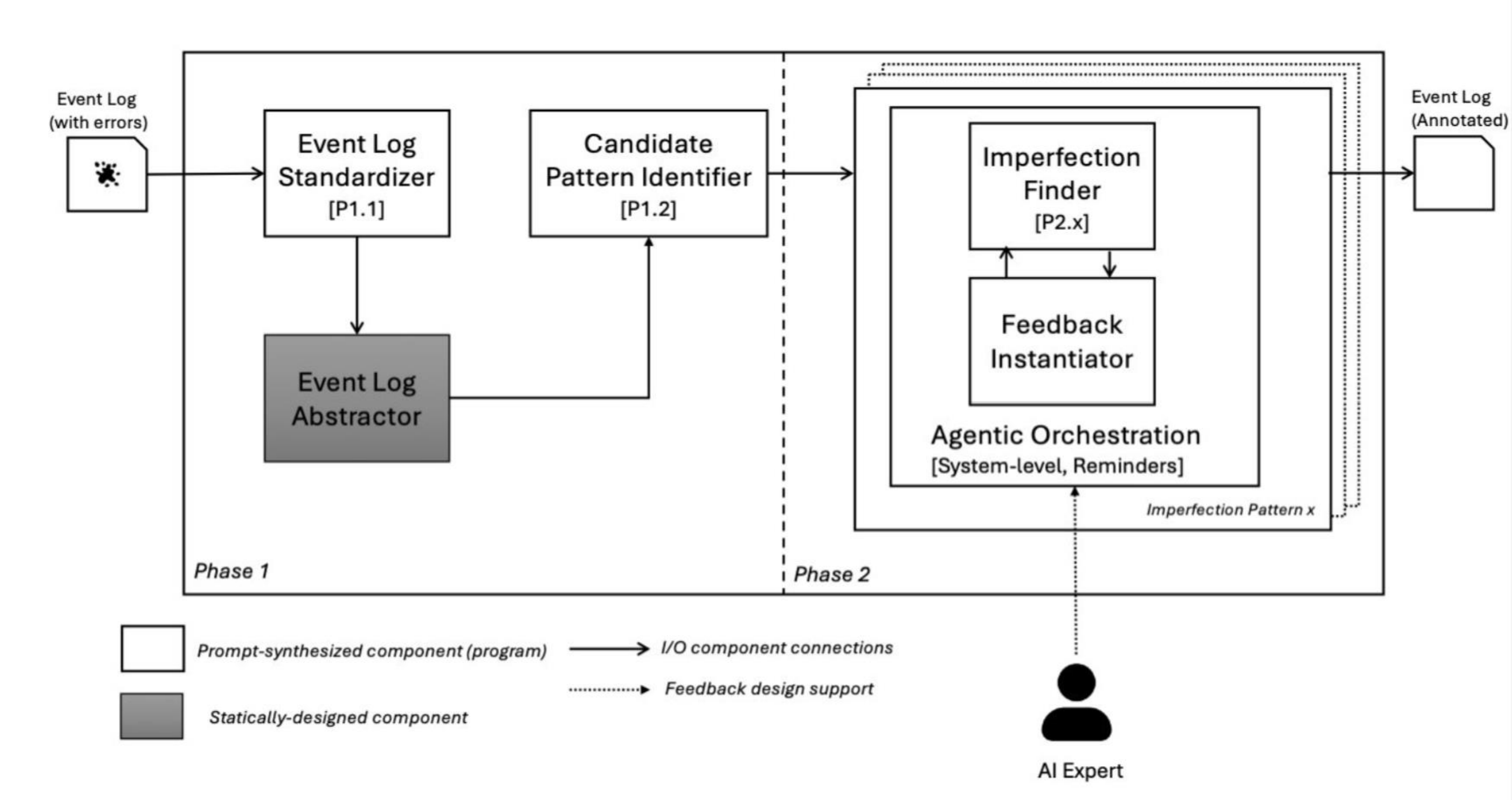
What if a Large Language Model (LLM) could do this automatically?

Vibe Coding

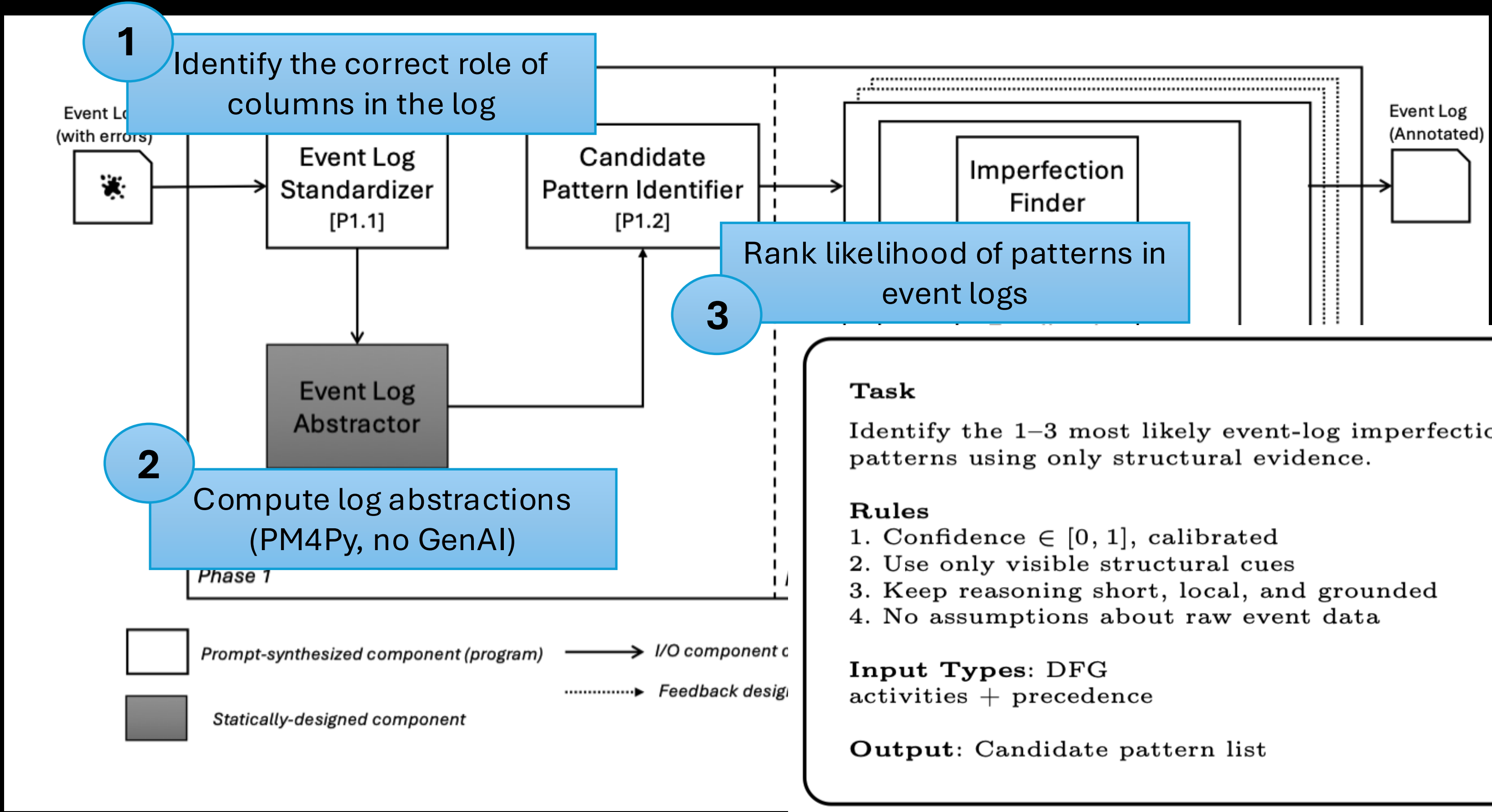


make it do
the thing... u know?
on it! here's
a first pass ✨
LLM
idea / prompt
suggestion
... and repeat until it vibes

Our Approach



Phase 1: Identify Candidate Patterns



Task

Identify the 1–3 most likely event-log imperfection patterns using only structural evidence.

Rules

1. Confidence $\in [0, 1]$, calibrated
2. Use only visible structural cues
3. Keep reasoning short, local, and grounded
4. No assumptions about raw event data

Input Types: DFG activities + precedence

Output: Candidate pattern list

Dataset for Experiment

Log	Events	Act. Labels	Traces	% Top-10 Act. Labels	% Top-10 Variants
BPIC11	29004	186	1026	47.3%	10.8%
BPIC15	33574	281	696	25.3%	4.8%
Credit	63980	12	5000	94.7%	74.2%
Pub	66524	12	5000	90.5%	94.5%

Real-world (BPIC) and synthetic event logs

Each type of error injected by FLAWD[1], affecting 2%, 10%, 30% of events

[1] GitHub - jonghyeonk/FLAWD · GitHub

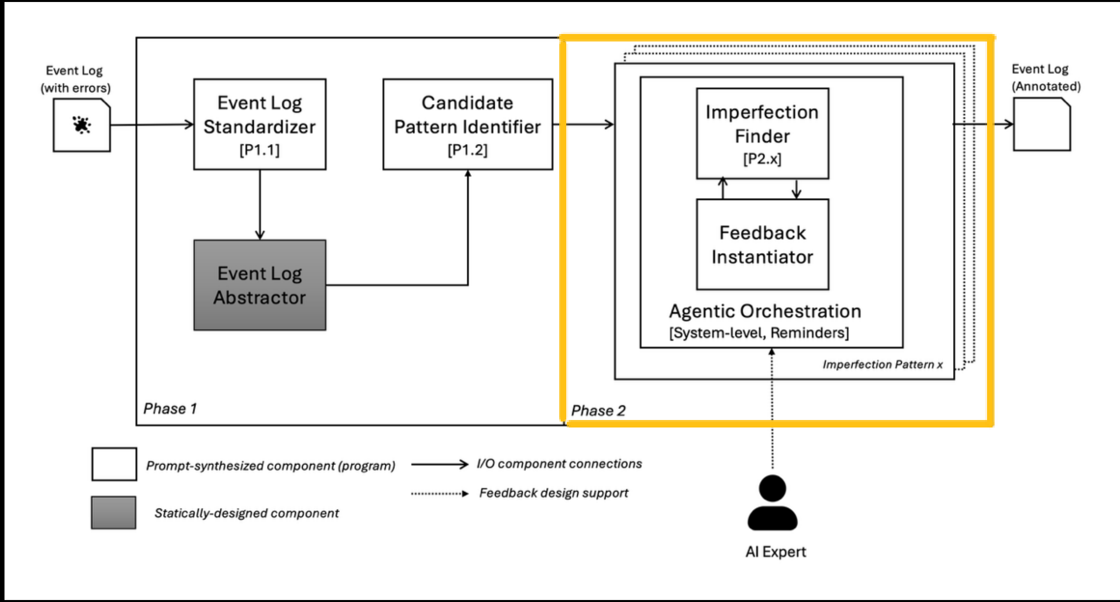
FORM-BASED
DISTORTED
HOMONYMOUS
SYNONYMOUS
COLLATERAL
POLLUTED

Data	Pattern	Ratio	DFG					Variant					WF-Net				
			GPT	Llama	DeepSeek	Grok	Avg	GPT	Llama	DeepSeek	Grok	Avg	GPT	Llama	DeepSeek	Grok	Avg
Credit	P1	0.02	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		0.10	0	0	0.5	0.5	0.25	0	0	0	0	0	0	0	0	0	0
		0.30	0	0	0	1	0.25	0	0	0	0	0	0	0	0	0	0
	P2	0.02	1	0	1	0	0.5	0	0	0	0.5	0.125	1	0	1	1	0.75
		0.10	1	0	1	1	0.75	0	0	0	0	0	1	0	1	0	0.5
		0.30	1	0	1	1	0.75	0.5	0	1	1	0.625	1	0	1	1	0.75
	P3	0.02	1	0	1	0	0.5	1	0.5	1	0	0.625	1	0	1	1	0.75
		0.10	1	0	1	1	0.75	1	0	1	1	0.75	1	0	1	1	0.75
		0.30	1	1	1	1	1	1	0	1	1	0.75	1	1	1	1	1
	P4	0.02	1	0	1	0	0.5	0	0	0.5	0	0.125	1	0	0.5	0.5	0.5
		0.10	0	0	0.5	0	0.125	0	0	0.5	0.5	0.25	0.5	0.5	0.5	0	0.375
		0.30	0.5	0	0.5	0	0.25	0	0	1	0.5	0.375	0.5	0	1	0.5	0.5
	P5	0.02	1	0	1	1	0.75	1	0	1	1	0.75	0	0	1	1	0.5
		0.10	1	0	1	1	0.75	1	0	1	1	0.75	1	0	1	1	0.75
		0.30	1	0	1	1	0.75	1	0	1	1	0.75	1	0	1	1	0.75
	P6	0.02	1	0.5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	0	0.5
		0.10	1	0.5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	1	0.75
		0.30	1	0.5	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	1	1
Pub	P1	0.02	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0
		0.10	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0.125
		0.30	0	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0	0	0
	P2	0.02	1	0	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	1	1	0.75
		0.10	1	0	1	1	0.75	0.5	0	0.5	1	0.5	1	0	1	1	0.75
		0.30	1	0	1	1	0.75	0.5	0	1	1	0.625	1	0	1	1	0.75
	P3	0.02	0	0	0	0	0	0	0.5	0	0	0.125	0	0	0	0	0
		0.10	0	0	0.5	0	0.125	0.5	0	0	0	0.125	0	1	0	0	0.25
		0.30	0	0	0	0	0	0.5	0	0	0	0.125	0	0.5	0	0	0.125
	P4	0.02	0	0	0.5	0	0.125	0	0	0	0	0	0	1	0	0	0.25
		0.10	0.5	0	1	0	0	0.25	0	0	0	0	0	1	1	1	0.75
		0.30	1	0	1	0	0.5	0	0	0	0	0	0	1	0.5	0	0.375
	P5	0.02	1	0	1	0	0.5	0	0	0	0	0	0	1	0	0.5	0.375
		0.10	1	0	1	1	0.75	1	0	1	1	0.75	1	0	1	0	0.5
		0.30	1	0	1	1	0.75	1	0	1	1	0.75	1	0	1	1	0.75
	P6	0.02	1	0.5	1	1	0.875	0	0.5	0	1	0.375	0.5	1	N/A	N/A	0.75
		0.10	1	0.5	N/A	0	0.5	1	0.5	1	1	0.875	1	0.5	N/A	N/A	0.75
		0.30	1	0.5	N/A	1	0.83	1	0.5	1	1	0.875	1	0.5	N/A	1	0.83
Average by abstraction			GPT	Llama	DeepSeek	Grok	Avg	GPT	Llama	DeepSeek	Grok	Avg	GPT	Llama	DeepSeek	Grok	Avg
			0.67	0.11	0.72	0.54	0.51	0.40	0.11	0.46	0.46	0.36	0.61	0.14	0.72	0.51	0.50

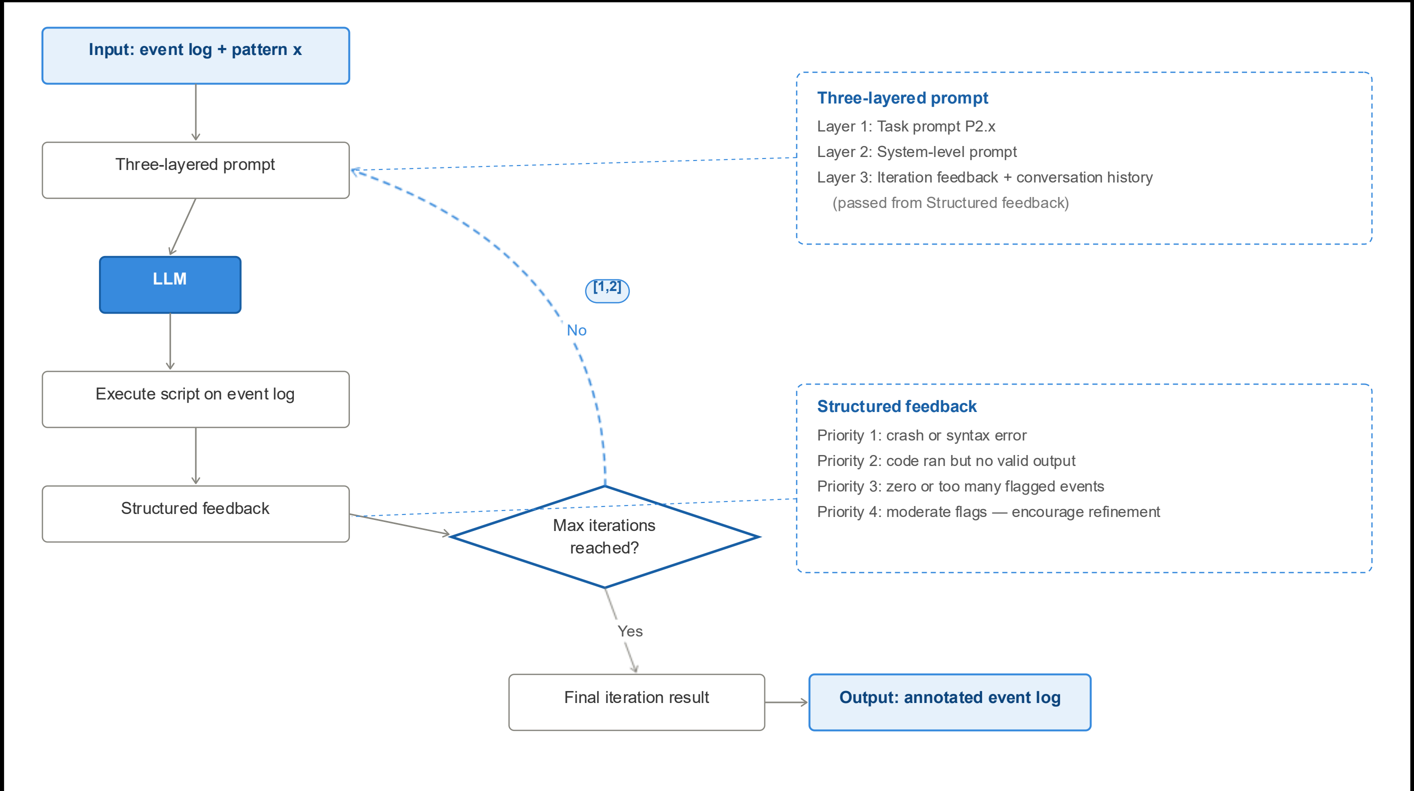
Process model abstractions better than list of variants

Phase 2

Can we automatically find exactly which events are responsible?

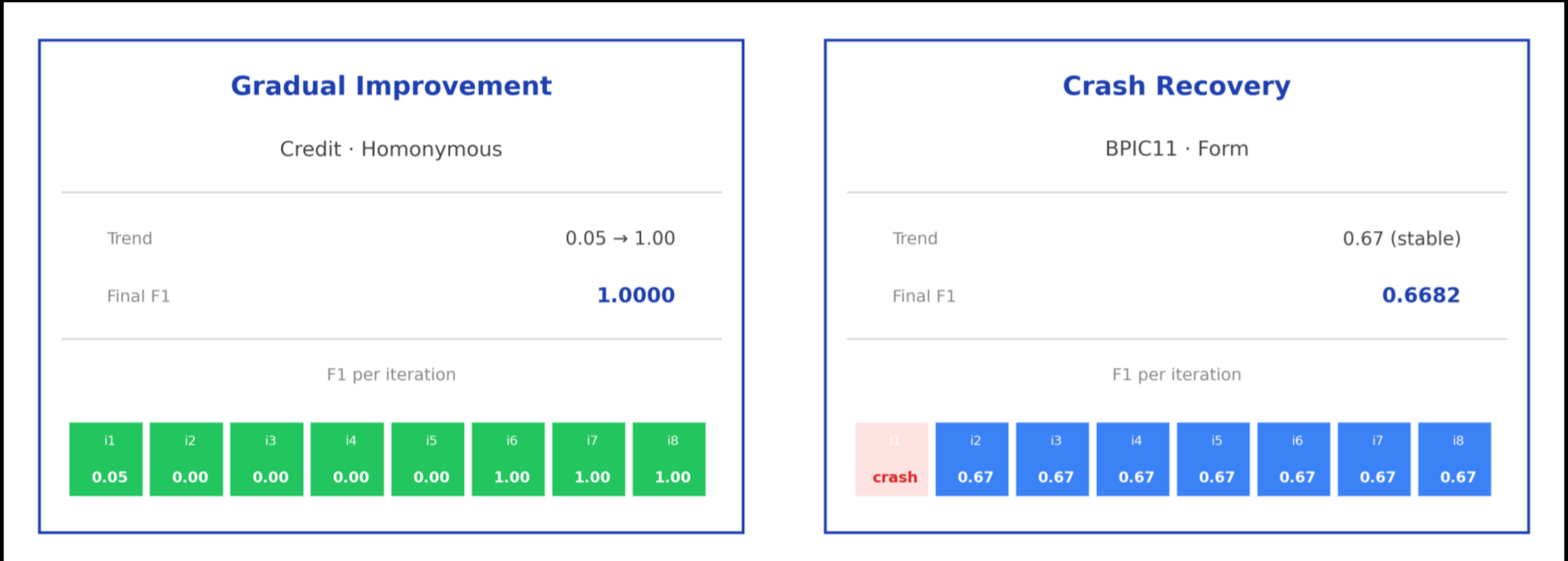


Phase 2: Agentic Feedback Loop

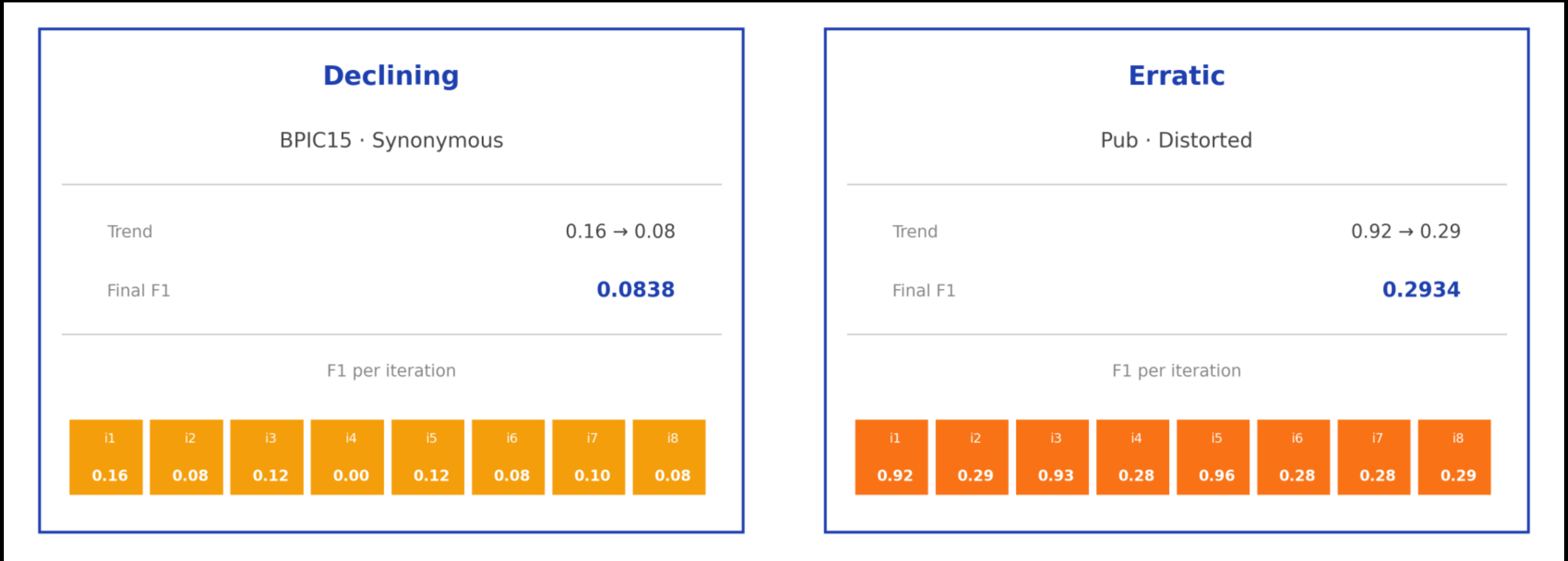


[1] Madaan, A., et al. (2023). *Self-Refine: Iterative Refinement with Self-Feedback*. NeurIPS 2023.
[2] Shinn, N., et al. (2023). *Reflexion: Language Agents with Verbal Reinforcement Learning*. NeurIPS 2023.

Trajectory examples: GI and CRS



Trajectory examples: Declining and Erratic



To conclude…

Predictive AI to predict ”aspects of interests” of business process future execution

For Generative AI: pick your use case, the sky is the limit!

THANKS!

https://marco-comuzzi.github.io

http://iel.unist.ac.kr/

@dr_bsad

mcomuzzi@unist.ac.kr