Predictive Process Monitoring for Prediction of Remaining Cycle Time in Automated Manufacturing

A Case Study

Friederich, Jonas; Lindeløv, Jonas Kristoffer; Lazarova-Molnar, Sanja

Published in:
2023 IEEE 28th International Conference on Emerging Technologies and Factory Automation (ETFA)

DOI:
10.1109/ETFA54631.2023.10275361

Publication date:
2023

Document version:
Accepted manuscript

Predictive Process Monitoring for Prediction of Remaining Cycle Time in Automated Manufacturing: A Case Study

Jonas Friederich
Mærsk Mc-Kinney Møller Institute
University of Southern Denmark
Odense, Denmark
jofr@mmmi.sdu.dk

Jonas Kristoffer Lindeløv
Kamstrup A/S
Skanderborg, Denmark
jokl@kamstrup.com

Sanja Lazarova-Molnar
Institute AIFB
Karlsruhe Institute of Technology
Karlsruhe, Germany
lazarova-molnar@kit.edu

***Abstract*—Predicting remaining cycle times of products in manufacturing systems is critical to ensure on-time deliveries to customers, schedule resources and actions for expected order completions, and address excessive production stops proactively rather than retroactively. Recent advances in Predictive Process Monitoring (PPM), a sub-discipline of Process Mining, enable the use of machine learning to predict remaining cycle times based on event data. We apply PPM to the automated manufacturing domain and demonstrate the approach using a case study from a water meter manufacturer. For prediction of remaining cycle times, PPM relies on regression methods, such as Decision Trees, Random Forests, and Gradient Boosting Machines based on event data. We compare the prediction accuracy of these methods and show that PPM can deliver relevant insights for production lines without imposing extensive data requirements.**

***Index Terms*—process mining, predictive process monitoring, remaining cycle time prediction, manufacturing systems**

I. Introduction

Manufacturing systems encompass processes that transform materials into finished products. In this context, a manufacturing process is a sequence of activities, each characterized by a duration and a set of resources. The total time required to manufacture a product is called cycle time. Cycle time is the basis for a significant number of planning, control and measurement activities in manufacturing environments, such as maintenance planning, production capacity planning and due date assessment [1].

Cycle time of a product depends on many different production variables, such as utilized resources or activity durations. Thus, cycle time prediction implies identifying these variables and building models for predicting the remaining cycle time. Proposed methods for cycle time prediction use, for example, Machine Learning and/or Process Mining techniques [2].

Process Mining (PM) is a process management technique that enables the reconstruction and evaluation of business processes based on digital traces, for example of products, in information systems. PM, thus, facilitates to model the implicit and otherwise hidden process knowledge contained in data to make such knowledge tangible and transportable [3]. PM techniques usually inform about cases that are already closed and cannot be changed (i.e., *post-mortem event logs*). More recently, a new sub-discipline of PM has emerged, namely Predictive Process Monitoring (PPM), which is concerned with predicting the future of an uncompleted process execution [4]. PPM, thus, takes into account *pre-mortem event logs* that relate to cases that are not yet closed. If a case is still ongoing, we can exploit current case-related event data to ensure the correct and efficient processing of that case for example, by predicting whether that case will be closed in due time.

In this paper, we apply PPM to the manufacturing domain to predict the remaining cycle time of production orders in manufacturing systems. We evaluate the approach on a real industrial dataset from a water meter manufacturer. For this, we calculate custom features such as the time since the last event, extract partial traces, encode these enriched partial traces using state-of-the-art encoding methods, and train regression algorithms such as Decision Trees, Gradient Boosting Machines and K-Nearest Neighbors on the encoded data. We, furthermore, evaluate the regression results using well-established error metrics, such as Mean Absolute Error and Root Mean Squared Error. We show that PPM can be used to accurately predict remaining cycle times in manufacturing environments without extensive data requirements.

The remainder of this paper is organized as follows. In Section 2, we present the background on Predictive Process Monitoring. Section 3 covers the approach for remaining cycle time prediction in manufacturing using PPM. In Section 4, we apply the approach to a case study and report the results. We review related work in Section 5 and provide a summary and an outlook in Section 6.

II. Background on Predictive Process Monitoring

Production processes are subject to internal policies, best practices, standards, regulations and laws. For this reason, monitoring such processes to ensure that production operates within these constraints is critical in many organizations. However, many process monitoring approaches are reactive and detect violations merely after they have occurred. In contrast, PPM, a subdiscipline of Process Mining and Predictive

Analytics, can provide early insights so that users can manage ongoing process executions to meet business constraints. Instead of just detecting violations, they are predicted and can potentially be prevented [4], [5].

PPM, introduced in 2014 by Maggi et al. [4], deals with predicting the future of an uncompleted process execution. Predictions target the outcome of an execution of a process (i.e., case), its completion time, or the sequence of its future activities. Approaches for PPM typically learn from complete historical cases to make predictions about the future of an ongoing and uncompleted case. As typical for supervised machine-learning-based approaches, PPM approaches also feature two phases: a *training phase* and a *prediction phase*. In the training phase, a predictive model is learned from historical traces, and in the prediction phase, the trained model is used to predict the future of an ongoing trace [6].

Predictions often depend on both the sequence of activities (i.e., control-flow) executed in a given case, and the values of data attributes (i.e., payload) after each activity in a case [4]. For example, a manufacturing system may produce different types of products. Each product is characterized by a specific production sequence, utilizing different machinery. To make accurate predictions about ongoing traces, we need to consider the production sequence and attributes such as *product type* and *utilized machinery*.

Data attributes can be either static or dynamic. Static attributes are case-dependent and do not change over time. Dynamic attributes, on the other hand, can change after the execution of an activity. Consider the example attributes above: The *product type* will not change over time in most scenarios, while the *utilized machinery* for a production activity will likely change over time.

According to Di Francescomarino & Ghidini [6], approaches for PPM can be roughly divided into three main dimensions *prediction type*, used *approach* and used *input data* (Table I). In the following, we describe these three dimensions in more detail.

Prediction types usually fall into one of the following three categories: *outcome-based predictions* (i.e., predicting categorical or Boolean outcome values from a fixed set), *numeric value predictions* (i.e., predicting numeric measures of interest), or *next event predictions* (i.e., predicting future events and their corresponding payloads). For example, in manufacturing, predicting an outcome could be whether a product will be discarded or require rework; predicting a numerical value could be the remaining cycle time of a production order; and predicting next events could include the remaining path a product will take through the manufacturing system.

Approaches used for PPM can be divided into two main categories: *explicit model-based approaches* (i.e., approaches relying on an explicit process model) and *supervised-learning-based approaches* (i.e., approaches utilizing supervised machine learning, e.g., classification and regression models).

Finally, the input data used for PPM can be classified into four main categories: *control-flow* (i.e., sequence of events), *event payload* (i.e., payload associated to the events, e.g., timestamps or resources), *unstructured information* (i.e., textual information available together with the event log), or *contextual information* (i.e., information related to process context, e.g., resource availability).

In this article, we apply supervised machine-learning-based approaches to predict the remaining cycle time of ongoing production orders in a production line. We do so by utilizing control flow and event payload input data, such as product type, batch size and shift type.

III. Predictive Process Monitoring for Cycle Time Prediction in Manufacturing

In this section, we describe the steps of remaining cycle time prediction of ongoing traces using supervised machine learning. Figure 1 provides the high-level workflow of the approach. Sensors in modern manufacturing systems generate data about the operation of the system. This data is then distributed to enterprise information systems such as Manufacturing Execution Systems (MES) which may store the data for traceability and analysis. From these records, we extract historical and ongoing traces capturing information about the production of individual production orders. Based on the historical traces, we train a machine learning model that is used to predict the remaining cycle times of ongoing traces. At inference, the trained model can be used to make predictions about ongoing traces.

In the following two subsections, we first describe the main steps to train a machine learning model for remaining cycle time prediction (Subsection III-A) and then present measures to evaluate a trained model (Subsection III-B).

A. Model Training

Let $\mathcal{E}$ be the universe of events and let $\sigma \in \mathcal{E}^*$ be a trace. A trace records the execution of an instance of a process and is a finite sequence of events. Each event $e \in \mathcal{E}$ is unique and contains a timestamp. Let L be an event log defined as a set of traces $L \subseteq \mathcal{E}$. Given an event log L and an ongoing execution trace $\sigma_i^m = \langle e_1, ..., e_m \rangle$ of length m corresponding to the

TABLE I
Predictive Process Monitoring taxonomy (based on [6]). The characteristics of our case study are highlighted.

Dimension	Categories			
Prediction type	outcome	numeric	next events	
Approach	explicit model		supervised learning	
Input data	control flow	event payload	unstructured information	contextual information

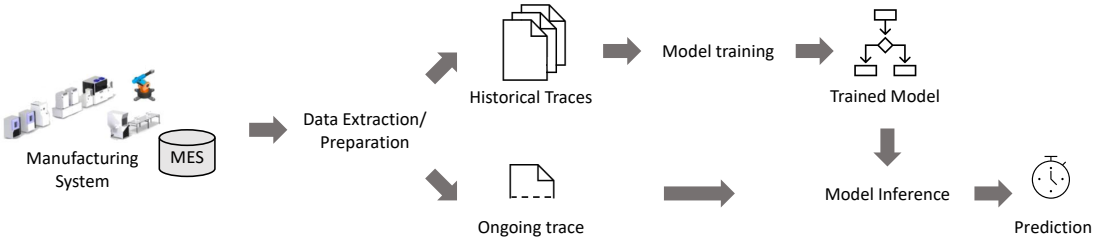

Fig. 1. Remaining cycle time prediction for manufacturing systems using PPM.

case with the identifier i, a function $f_n(L, \sigma_i^m)$ is learned that returns a predicted numerical value $\hat{y}_i$, which approximates the actual value y_i [3], [6]. In our case, y_i is the remaining cycle time of a production order. In the following, we describe the main steps for training a machine learning model for remaining cycle time prediction based on an event log.

1) Prefix Extraction: The trained model makes predictions on incomplete traces (i.e., prefix: what has happened; in contrast to suffix: what will happen) during inference. Therefore, systematic relationships between incomplete traces and the target variable (i.e., remaining cycle time) have to be learned in the training phase [6]. To learn these relationships, we create subsets of all extracted traces that include traces up to a given length. For example, consider a complete trace capturing six events. We can create up to five prefixes: the partial trace after the first event was executed, the partial trace after the first and the second event was executed, and so on. Using all possible prefixes might result in two problems: 1) a large number of prefixes significantly slows down the training of the predictive models, and 2) the predictive model might become biased towards longer cases. Thus, it is common to consider prefixes up to a certain number of events [7].

2) Encoding: Based on the extracted prefixes, prefix traces and labels are encoded in the form of fixed-length feature vectors. Such vectors can be processed by supervised machine learning techniques and are used to train a predictive model from the encoded data [6]. Features extracted from a trace may encode information about the performed activities during the execution of a trace and the activity sequence (i.e., control-flow features), as well as features corresponding to the payload of an event (i.e., payload features) [7]. In summary, we can encode each prefix trace σ_i^m and its corresponding label y_i (i.e., remaining cycle time) to a feature vector of the form $x_i^m = (x_{i1}, x_{i2}, ..., x_{im}, y_i)$. Table II displays the most commonly used encoding methods in existing PPM approaches that resort to both control flow and payload features [6], [7].

Static encoding uses one-hot encoding to encode static categorical data attributes and leaves numeric features as is. *Last payload encoding* considers dynamic data attributes and encodes them only for the last event in a trace. *Aggregated encoding* considers all events in a trace, but ignores their order. This allows multiple aggregation functions to be applied to the values of dynamic attributes of a case. *Complex index encoding* uses all available information and generates one feature for each event attribute per event executed. In this way, lossless encoding is achieved, allowing the original trace to be recovered from its feature vector [7]. In our case study, we use static encoding in combination with the other encoding techniques.

3) Supervised Learning Algorithm: The prediction of the remaining cycle time is a regression task. Popular algorithms for regression tasks are Decision Tree (DT), Random Forest (RF), k-Nearest Neighbors (k-NN) and gradient boosting machine (GBM) as well as extensions of GBM such as XGBoost. DTs are beneficial in terms of interpretability of results. RFs and GBMs are powerful ensemble machine learning algorithms that generally achieve better predictive accuracy than a single DT, but at the cost of interpretability. k-NN is a simple but powerful algorithm with few hyperparameters to tune. However, k-NN suffers from large computation cost during inference.

TABLE II
TRACE ENCODING METHODS [7].

Encoding technique	Attribute types	Trace abstraction	Feature extraction	
			Numeric	Categorical
Static	Static	Case attributes	as is	one-hot
Last payload	Dynamic	Last event	as is	one-hot
Aggregation	Dynamic	All events, unordered	min, max, mean, sum, std	frequencies or occurrences
Complex index	Dynamic	All events, ordered	as is (for each index)	one-hot (for each index)

B. Model Evaluation

Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are two of the most common error metrics used to measure the accuracy of regression tasks. MAE measures the average magnitude of errors in a set of predictions. It is the average of the absolute differences between prediction y_i and actual observation $\hat{y}_i$ in a test sample n, with all individual differences equally weighted.

$$MAE = \frac{1}{n}\sum_{j=i}^{n}|y_i - \hat{y}_i| \tag{1}$$

RMSE is a quadratic scoring rule that also measures the average magnitude of the error. It is the square root of the average of the squared differences between prediction and actual observation.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{2}$$

Tax et al. [2] and Wahid et al. [8] argue that the time differences between events is heavy-tailed with many orders of magnitudes differences compared to, for example, a uniform or a normal distribution. RMSE is overly sensitive to such occasional extreme gaps between events due to the quadratic cost function in RMSE compared to the identity cost function in MAE. We use MAE as the main error metric because we are predicting time-difference data and because the negative utility prolonged cycle times is linear with time (each additional second lost has a fixed cost) - not quadratic as for RMSE.

IV. Case Study

In our case study, we apply PPM to the manufacturing domain by training several supervised machine learning models on a dataset provided by a case company to predict the remaining cycle time of ongoing production orders. We, furthermore, empirically evaluate the trained models and compare their accuracies.

A. Case Company

The case company, Kamstrup A/S, is a manufacturer of smart metering solutions for electricity, water, and district heating. Kamstrup produces millions of meters annually for customers on all continents. The company has an interest in predicting the remaining cycle time of products on their manufacturing lines as part of an ongoing effort to maximize production efficiency. Real-time predictions will be used to alert staff about upcoming production stops, thereby decreasing response time and increasing uptime, in contrast to the current purely reactive strategy. Furthermore, interpretation of regression models and their calculated feature importances (e.g., Gini Importance for DTs) can provide insights about what features are risk factors for production inefficiencies, thus, constituting a data-driven input on where to focus optimization activities.

Our case study is based on a particular assembly process at one of the company's automated water meter production lines, which is a critical process as it represents a bottleneck in the overall production of water meters. In this assembly process, a series of fully automated industrial robots mount a measuring tube, a printed circuit board (PCB), and protective/structural utilities into the water meter housing. The mounting process consists of 8 sequential assembly activities. At full utilization, production is scheduled into three shifts (i.e., day, evening, night) during weekdays (i.e., Monday-Friday). Most staff is permanently employed at a particular shift. Customers can choose from of a variety of meter types and order sizes are typically in the range from a few hundred to a few thousand meters. The meters are manufactured in batches of the same meter type. All of this information is encoded as payload attributes into the event log.

To understand the nature of the production, the dotted chart in Figure 2 visualizes the production schedule on a ten-day extract. The vertical axis depicts the cumulative number of produced meters and the horizontal axis the time. Each dot represents the start of a new production order. The dots are colored by shift type. We can see that there was no production during the weekend and that production occasionally stops for maintenance or changeover work.

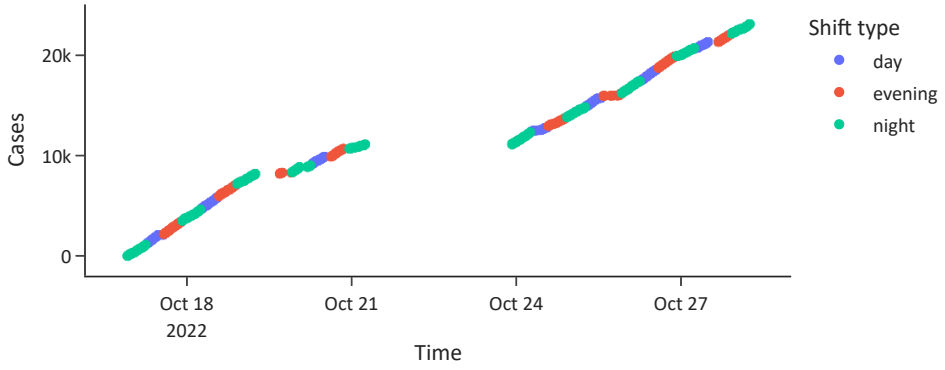

Fig. 2. Dotted chart of the production process.

B. Experiment Setup

We constructed an event log from two months of historical data that was extracted from Kamstrup production databases used by the MES system. The log is a complete record of all production process executions in the period. We excluded production orders that were not completely contained in the observation period, i.e., both the first and the last activity had to be performed within the observation period. In addition, after consultation with the process owner, we removed less frequent activities that are not related to the main production process. This resulted in a final event log containing 115,686 production orders (i.e., cases) that were processed through 925,488 recorded events in 81 batches.

The event log records the previously described meter housing mounting process. Table III shows an excerpt of the event log consisting of a production order identifier, activity name, timestamp, meter type, batch size and shift type. *Meter type* is a static categorical feature, *batch size* a static numerical feature (i.e., case dependent) and *shift type* a dynamic categorical feature. The timestamp captures the start time of the respective activity.

Figure 3 shows a histogram and a box plot representing the recorded order cycle times. As can be seen from the histogram, there are several outliers that the company is particularly interested in predicting. The median cycle time is 110 seconds, the lower quartile is 102 seconds and the upper quartile is 128 seconds.

During preprocessing of the event log, we designed seven dynamic numerical features based on the existing timestamp feature, corresponding to the *time since the start of a production order* (TSS), *time since the last event* (TSLE), *event number* in the trace, *time since midnight* (TSM), as well as current *month*, *weekday* and *hour*. Furthermore, we calculated the *remaining cycle time* (RCT) for each of the data samples - the ground truth for the machine learning model. In Table IV, we show exemplary values for the features TSM, TSS, TSLE and the remaining cycle time label for one production order.

For each of the experiments, we encoded the data using either the *last payload*, *aggregation*, *complex index* or a combination of the former two methods. We, further, randomly split the dataset into training and testing datasets in a ratio of 80:20. The trained regression methods were tree-based (ordinary Decision Tree, Random Forest and XGBoost) and k-Nearest Neighbors. For each of the methods, we performed basic hyperparameter optimization using grid search. The hyperparameters tested for each regression method as well as the considered parameter range is shown in Table V. We compared the methods against a baseline model, predicting the mean remaining cycle time for each production order independent of the prefix length of the input trace. We, furthermore, compared the methods against an informed baseline model predicting the mean remaining cycle time for each production order based on the prefix length of the input trace. We implemented the case study using the *Nirdizati Predictive Process Monitoring*

TABLE III
SAMPLE EXCERPT OF THE EXTRACTED EVENT LOG.

Order ID	Activity	Timestamp	Meter Type	Batch Size	Shift Type
1	A	2022-08-31 22:00:08	Type 1	1344	Night
2	A	2022-08-31 22:00:24	Type 1	1344	Night
1	B	2022-08-31 22:00:28	Type 1	1344	Night
3	A	2022-08-31 22:00:41	Type 1	1344	Night
2	B	2022-08-31 22:00:45	Type 1	1344	Night
1	C	2022-08-31 22:00:53	Type 1	1344	Night
...	...	...	...	...	...
5434	A	2022-09-01 06:00:00	Type 4	333	Day
5433	B	2022-09-01 06:00:04	Type 4	333	Day
5430	E	2022-09-01 06:00:07	Type 4	333	Day
5431	D	2022-09-01 06:00:08	Type 4	333	Day
...	...	...	...	...	...

TABLE IV
EXEMPLARY LOG WITH DESIGNED FEATURES AND LABELS (TSM = *time since midnight* IN MINUTES; TSS = *time since start* IN SECONDS; TSLE = *time since last event* IN SECONDS; RCT = *Remaining cycle time* IN SECONDS).

Activity	Timestamp	...	TSM	TSS	TSLE	Event #	RCT
A	2022-08-31 22:00:08	...	1320	0	0	1	118.36
B	2022-08-31 22:00:28	...	1320	20.02	20.02	2	98.34
C	2022-08-31 22:00:53	...	1320	44.47	24.45	3	73.89
D	2022-08-31 22:01:04	...	1321	56.15	11.68	4	62.21
E	2022-08-31 22:00:08	...	1321	69.49	13.34	5	48.87
F	2022-08-31 22:01:27	...	1321	78.78	9.29	6	39.58
G	2022-08-31 22:01:42	...	1321	93.57	14.79	7	24.79
H	2022-08-31 22:02:06	...	1322	118.36	24.79	8	0

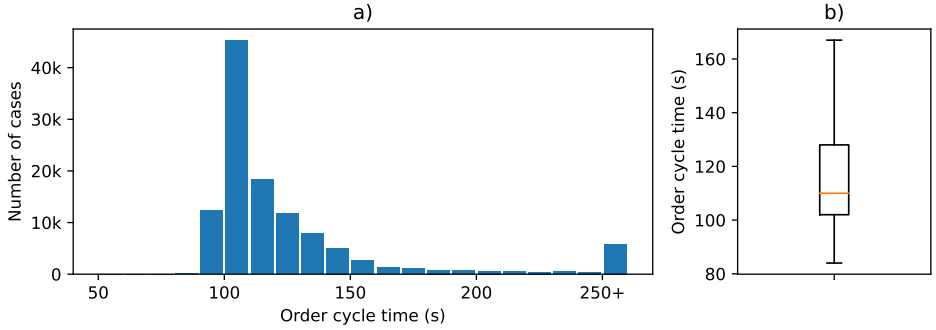

Fig. 3. Histogram (a) and box plot (b) describing the order cycle times. For better readability, outliers are not visualized in the box plot.

TABLE V
TESTED HYPERPARAMETERS FOR EACH REGRESSION METHOD.

Method	Hyperparameter	Description	Tested range
Decision Tree	max_depth	max. tree depth	[1,10]
	$max\_features$	fraction of features to consider for split	[0.3,1.0]
Random Forest	$n\_estimators$	number of estimators/trees	[10,500]
	max_depth	max. tree depth	[1,10]
	$max\_features$	fraction of features to consider for split	[0.3,1.0]
XGBoost	$n\_estimators$	number of estimators/trees	[10,500]
	max_depth	max. tree depth	[1,10]
	$max\_features$	fraction of features to consider for split	[0.3,1.0]
	$colsample\_bytree$	fraction of features used in each tree	[0.3,1.0]
	$subsample$	fraction of samples used in each tree	[0.3,1.0]
	$learning\_rate$	learning rate of the estimator	[0.001,0.1]
k-Nearest Neighbors	$n\_neighbors$	number of neighbors to use	[100,10000]

Engine [9].

C. Results

We measured the performance of the selected regression methods and compared the regression results using MAE and RMSE. As justified in Section III-B, we consider MAE as the main error metric. The final results are listed in Table VI. All trained models perform better than the baseline model and on par with the informed baseline model, with the exception of XGBoost, which performs slightly better.

The graphs in Figure 4 plot the MAE for each prefix length using the different encoding methods. For each of the encoding methods, the trained models achieve similar results in terms of MAE. As expected, on shorter prefixes the error is higher, and consistently decreases towards an increased prefix length. This observation can be explained by the assumption that the fewer activities, and thus less cycle time remain, the fewer sources of unpredictable variability remain in the data.

V. RELATED WORK

This section reviews existing approaches for remaining cycle time prediction in the field of PPM, as well as related work on remaining cycle time prediction within the manufacturing domain. To the best of our knowledge, there is no contribution investigating the applicability of PPM in the manufacturing domain.

A. Domain-independent Approaches for Remaining Cycle Time Prediction using Predictive Process Monitoring

As described in Section II, PPM approaches can be divided in two main categories: *explicit model-based approaches* as well as *supervised-learning-based approaches*. Below, we review approaches for each of these categories.

Model-based approaches use an explicit process model discovered from an event log. This model is further enriched with remaining time information for each activity extracted from historical traces. This enriched model can be used for prediction of incomplete traces [6]. Transition systems, a modeling formalism to describe the potential behavior of discrete systems, are mainly used for model based-approaches. One of the early approaches using transition systems for remaining time prediction is the one by van der Aalst et al. [10]. Polato et al. extend their approach by enriching the transition systems using machine learning models such as Naïve Bayes and Support Vector Regression [11] or by also taking into account data payloads [12]. Instead of transition systems, some contributions use sequence trees [13] to identify partial process models using sequential pattern mining or stochastic Petri nets [14] as explicit models to predict the remaining time of a process instance.

Supervised-learning-based approaches generally use the pipeline described in Section III-A to learn models that are able to predict the remaining time of an ongoing process execution. The necessary steps include prefix extraction, trace encoding and model training. Approaches can be distinguished

TABLE VI
OVERVIEW OF RESULTS.

Encoding type	Last payload		Aggregation		Complex index		Last payload & aggregation	
Regression Method	MAE	RMSE	MAE	RMSE	MAE	RMSE	MAE	RMSE
Baseline	43.05	166.84	43.05	166.84	43.05	166.84	43.05	166.84
Informed baseline	28.24	162.92	28.24	162.92	28.24	162.92	28.24	162.92
Decision Tree	27.92	163.00	27.79	162.79	28.06	166.52	27.87	164.23
Random Forest	28.06	162.93	28.06	162.93	28.11	163.31	28.04	163.39
XGBoost	25.17	164.16	25.15	164.12	25.18	164.15	25.17	164.20
k-Nearest Neighbors	28.23	162.92	27.78	163.38	28.05	163.06	27.72	163.34

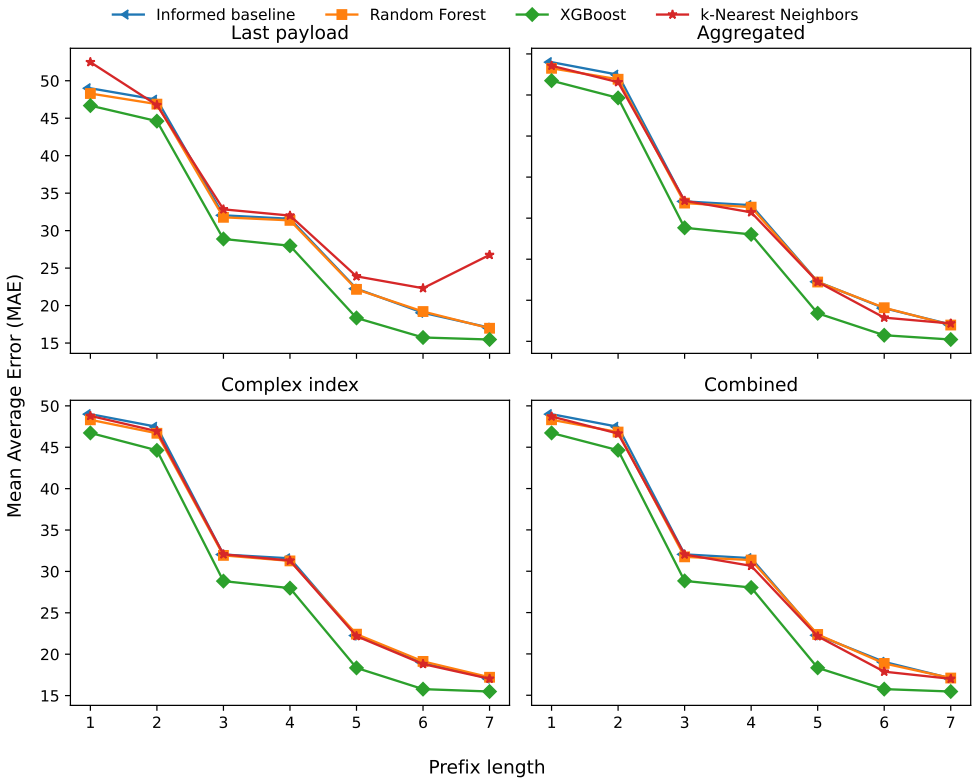

Fig. 4. MAE for each prefix length using different encoding methods. Baseline and Decision Tree are omitted for readability.

based the encoding technique and on the machine learning algorithm used. For example, de Leoni et al. [15] use aggregated and last state encoding of the traces and DTs for model training, and Verenich et al. [16] use complex index encoding and RFs. Tax et al. [2] and Wahid et al. [8] train more complex model architectures such as Long Short-Term Memory Neural Networks and Deep Neural Networks respectively.

Furthermore, a number of supervised-learning-based approaches suggest bucketing of trace prefixes. Then, for each bucket, one regression model is learned. For example, Leontjeva et al. [17] define buckets based on prefix length and Di Francescomarino [18] use clustering to assign traces to buckets.

Extensive reviews of the PPM field, covering all prediction types (i.e., numeric, outcome, and next events) can be found in Di Francescomarino et al. [5] and Márquez-Chamorro et al. [19].

B. Cycle Time prediction in Manufacturing Environments

Research on the prediction of product cycle times in manufacturing has a long history. Assessment of customer due dates, efficient resource scheduling and monitoring of operations require accurate cycle time predictions. For this important task, existing research utilizes both non-parametric [20] and parametric machine learning methods [1], [21], deep learning [22], probabilistic methods [23] and simulation modeling [24].

More specifically, Backus et al. [20] use clustering, K-nearest neighbors and Regression Trees to predict the remaining cycle time of wafer lots in a semiconductor manufacturing system. To predict lead times of a semiconductor manufacturer, Lingitz et al. [21] use several regression algorithms such as Ridge and Lasso. Choueiri et al. [1] use both a transition system and linear regression to create a hybrid model for remaining cycle time prediction in manufacturing. Fang et al. [22] present and approach that uses auto encoders to predict the cycle time in discrete manufacturing systems. An integration of both probabilistic (i.e., Bayesian network) and predictive models to enhance prediction results is presented by Ruschel et al. [23]. Chang & Liao [24] use simulation and a fuzzy-rule-based system to mimic the behavior of a semiconductor manufacturing system and to predict remaining cycle time.

VI. Summary and Outlook

We demonstrated the application of Predictive Process Monitoring (PPM) for remaining cycle time prediction in the manufacturing domain. We extracted an event log from a highly automated sequential production process of a water meter manufacturer, preprocessed the log, and applied and evaluated regression methods to it. The evaluation shows that we can achieve better or on par predictions of the remaining cycle time for the investigated water meter assembly process and regression methods than with a baseline model. We intend to publish the dataset that we used for our case study, which at the time of article submission is still being evaluated by Kamstrup A/S.

To further improve our work, we plan to conduct the following steps. Firstly, we aim to investigate the contribution of each feature to the prediction to provide better explainability of the results, which can help manufacturers like Kamstrup A/S take specific actions to improve process performance. Secondly, we will investigate the impact of obtaining end events in addition to the existing start events of activities for the studied production line. Knowing when an activity has actually finished instead of merely knowing the start of the next activity will allow for calculation of further features such as "cycle time of last activity". Thirdly, the application of bucketing as suggested in several related works may increase prediction accuracy. Fourthly, Senderovich et al. [25] state that both intra and inter-case features should be considered for accurate prediction of remaining cycle time. Intra-case features depend on the execution history of a particular case, e.g., duration between events or other case-specific attributes. Inter-case features, on the other hand, capture the interaction of all ongoing cases by clustering them into case types. For example, some orders may have a higher priority and thus do not need to compete over shared resources. Finally, we plan to reformulate the prediction problem into a binary classification task by labeling production orders and their respective traces either as outliers (e.g., cycle time greater than 250 seconds) or as regular cases. This could support the manufacturer's decisions in terms of due date assessment by providing easy to interpret predictions about unfinished production orders.

REFERENCES

[1] A. C. Choueiri, D. M. V. Sato, E. E. Scalabrin, and E. A. P. Santos, "An extended model for remaining time prediction in manufacturing systems using process mining," *Journal of Manufacturing Systems*, vol. 56, pp. 188–201, Jul. 2020.

[2] N. Tax, I. Verenich, M. La Rosa, and M. Dumas, "Predictive Business Process Monitoring with LSTM Neural Networks," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, E. Dubois and K. Pohl, Eds. Cham: Springer International Publishing, 2017, pp. 477–492.

[3] W. van der Aalst, "Data Science in Action," in *Process Mining: Data Science in Action*, W. van der Aalst, Ed. Berlin, Heidelberg: Springer, 2016, pp. 3–23.

[4] F. M. Maggi, C. Di Francescomarino, M. Dumas, and C. Ghidini, "Predictive Monitoring of Business Processes," in *Advanced Information Systems Engineering*, ser. Lecture Notes in Computer Science, M. Jarke, J. Mylopoulos, C. Quix, C. Rolland, Y. Manolopoulos, H. Mouratidis, and J. Horkoff, Eds. Cham: Springer International Publishing, 2014, pp. 457–472.

[5] C. Di Francescomarino, C. Ghidini, F. M. Maggi, and F. Milani, "Predictive Process Monitoring Methods: Which One Suits Me Best?" in *Business Process Management*, M. Weske, M. Montali, I. Weber, and J. vom Brocke, Eds. Cham: Springer International Publishing, 2018, vol. 11080, pp. 462–479.

[6] C. Di Francescomarino and C. Ghidini, "Predictive Process Monitoring," in *Process Mining Handbook*, ser. Lecture Notes in Business Information Processing, W. M. P. van der Aalst and J. Carmona, Eds. Cham: Springer International Publishing, 2022, pp. 320–346.

[7] I. Teinemaa, M. Dumas, M. L. Rosa, and F. M. Maggi, "Outcome-Oriented Predictive Process Monitoring: Review and Benchmark," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 2, pp. 17:1–17:57, Mar. 2019.

[8] N. A. Wahid, T. N. Adi, H. Bae, and Y. Choi, "Predictive Business Process Monitoring – Remaining Time Prediction using Deep Neural Network with Entity Embedding," *Procedia Computer Science*, vol. 161, pp. 1080–1088, Jan. 2019.

[9] K. Jorbina, A. Rozumnyi, I. Verenich, C. Di Francescomarino, M. Dumas-Menijvar, C. Ghidini, F. Maggi, M. La Rosa, and S. Raboczi, "Nirdizati: A web-based tool for predictive process monitoring," in *Proceedings of the BPM Demo Track and BPM Dissertation Award*, A. Kumar, M. Weske, R. Clariso, H. Leopold, J. Mendling, B. Pentland, and W. van der Aalst, Eds., 2017, pp. 1–5.

[10] W. van der Aalst, M. H. Schonenberg, and M. Song, "Time prediction based on process mining," *Information Systems*, vol. 36, no. 2, pp. 450–475, Apr. 2011.

[11] M. Polato, A. Sperduti, A. Burattin, and M. de Leoni, "Data-aware remaining time prediction of business process instances," in *2014 International Joint Conference on Neural Networks (IJCNN)*, Jul. 2014, pp. 816–823.

[12] M. Polato, A. Sperduti, A. Burattin, and M. d. Leoni, "Time and activity sequence prediction of business process instances," *Computing*, vol. 100, no. 9, pp. 1005–1031, Sep. 2018.

[13] M. Ceci, P. F. Lanotte, F. Fumarola, D. P. Cavallo, and D. Malerba, "Completion Time and Next Activity Prediction of Processes Using Sequential Pattern Mining," in *Discovery Science*, ser. Lecture Notes in Computer Science, S. Džeroski, P. Panov, D. Kocev, and L. Todorovski, Eds. Cham: Springer International Publishing, 2014, pp. 49–61.

[14] A. Rogge-Solti and M. Weske, "Prediction of Remaining Service Execution Time Using Stochastic Petri Nets with Arbitrary Firing Delays," in *Service-Oriented Computing*, ser. Lecture Notes in Computer Science, S. Basu, C. Pautasso, L. Zhang, and X. Fu, Eds. Berlin, Heidelberg: Springer, 2013, pp. 389–403.

[15] M. de Leoni, W. M. P. van der Aalst, and M. Dees, "A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs," *Information Systems*, vol. 56, pp. 235–257, Mar. 2016.

[16] I. Verenich, M. Dumas, M. La Rosa, F. M. Maggi, and C. Di Francescomarino, "Complex Symbolic Sequence Clustering and Multiple Classifiers for Predictive Process Monitoring," in *Business Process Management Workshops*, ser. Lecture Notes in Business Information Processing, M. Reichert and H. A. Reijers, Eds. Cham: Springer International Publishing, 2016, pp. 218–229.

[17] A. Leontjeva, R. Conforti, C. Di Francescomarino, M. Dumas, and F. M. Maggi, "Complex Symbolic Sequence Encodings for Predictive Monitoring of Business Processes," in *Business Process Management*, ser. Lecture Notes in Computer Science, H. R. Motahari-Nezhad, J. Recker, and M. Weidlich, Eds. Cham: Springer International Publishing, 2015, pp. 297–313.

[18] C. D. Di Francescomarino, M. Dumas, F. M. Maggi, and I. Teinemaa, "Clustering-Based Predictive Process Monitoring," *IEEE Transactions on Services Computing*, vol. 12, no. 6, pp. 896–909, Nov. 2019.

[19] A. E. Márquez-Chamorro, M. Resinas, and A. Ruiz-Cortés, "Predictive Monitoring of Business Processes: A Survey," *IEEE Transactions on Services Computing*, vol. 11, no. 6, pp. 962–977, Nov. 2018.

[20] P. Backus, M. Janakiram, S. Mowzoon, C. Runger, and A. Bhargava, "Factory cycle-time prediction with a data-mining approach," *IEEE Transactions on Semiconductor Manufacturing*, vol. 19, no. 2, pp. 252–258, May 2006.

[21] L. Lingitz, V. Gallina, F. Ansari, D. Gyulai, A. Pfeiffer, W. Sihn, and L. Monostori, "Lead time prediction using machine learning algorithms: A case study by a semiconductor manufacturer," *Procedia CIRP*, vol. 72, pp. 1051–1056, Jan. 2018.

[22] W. Fang, Y. Guo, W. Liao, K. Ramani, and S. Huang, "Big data driven jobs remaining time prediction in discrete manufacturing system: a deep learning-based approach," *International Journal of Production Research*, vol. 58, no. 9, pp. 2751–2766, May 2020.

[23] E. Ruschel, E. d. F. Rocha Loures, and E. A. P. Santos, "Performance analysis and time prediction in manufacturing systems," *Computers & Industrial Engineering*, vol. 151, p. 106972, Jan. 2021.

[24] P. C. Chang and T. W. Liao, "Combining SOM and fuzzy rule base for flow time prediction in semiconductor manufacturing factory," *Applied Soft Computing*, vol. 6, no. 2, pp. 198–206, Jan. 2006.

[25] A. Senderovich, C. Di Francescomarino, C. Ghidini, K. Jorbina, and F. M. Maggi, "Intra and Inter-case Features in Predictive Process Monitoring: A Tale of Two Dimensions," in *Business Process Management*, ser. Lecture Notes in Computer Science, J. Carmona, G. Engels, and A. Kumar, Eds. Cham: Springer International Publishing, 2017, pp. 306–323.