

The 8th International Conference on Emerging Data and Industry (EDI40),
April 22–24, 2025, Patras, Greece

A semi-automated approach to detecting process-level activities from sensor data

Luciano García-Bañuelos^{a,*}, Mauricio Jacobo González González^a, Ronny Seiger^b, Marco Franceschetti^b, Alejandra Guadalupe Silva Trujillo^c

^a*Department of Computer Science, Tecnológico de Monterrey, Puebla, Mexico*

^b*Institute of Computer Science, University of St.Gallen, St. Gallen, Switzerland*

^c*Universidad Autonoma de San Luis Potosi, San Luis Potosi, Mexico*

Abstract

Internet of Things (IoT) technologies can be leveraged to monitor and check the conformance of business process executions in the absence of a Business Process Management (BPM) System. This poses the challenge of detecting process activities from *low-level* sensor data by means of event abstraction. Existing methods rely on fully supervised approaches, combining sensor data and process event logs to train classification models. However, frequently these logs are not available, invalidating the applicability of such methods. In this paper, we propose a semi-automated approach to detecting process activities from sensor data based on frequently repeated subsequences and automata. We evaluate the approach with a proof-of-concept implementation and data from a small-scale smart factory. Our evaluation demonstrates the applicability of the proposed approach and its effectiveness in detecting process activities from sensor data to support domain experts with data analysis through automated suggestions.

© 2025 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the Conference Program Chairs.

Keywords: Internet of Things; Business Process Management; Activity detection; Cyber-physical Systems; Sensors; Industry 4.0

1. Introduction

The ongoing developments of Internet of Things (IoT) and Cyber-Physical Systems (CPS) bridge the physical and the digital world, allowing the monitoring of events happening in the physical world and generating their digital representations [5]. Here, the infusion of Business Process Management (BPM) technologies and BPM-related data constitutes an opportunity for managing CPS in a process-oriented way, in particular by leveraging conformance checking and other sophisticated process mining techniques [16]. This requires the availability of process *event logs* describing executions with process semantics, i.e., describing sequences of process activities [30]. However, BPM

* Corresponding author.

E-mail address: luciano.garcia@tec.mx

systems orchestrating and monitoring the execution of processes are typically not available in preexisting CPS, which prevents the generation of event logs with such process semantics [27]. IoT technologies are a solution to integrate these aspects [16]. These IoT-augmented systems offer sensor data that provides a low-level description of events happening in the physical world. This low-level data needs to undergo an *event abstraction* step translating it into high-level process events to enable process-oriented analyses [27]. Multiple attempts have been made by prior research to tackle the challenge of detecting process activities from sensor data [4]. However, these works typically assume a high degree of domain expert involvement to identify and label events and activities. Therefore, state-of-the-art approaches to activity detection from sensor data are usually fully supervised. This puts the burden on domain experts, which makes the activity detection error-prone and inefficient as it is difficult for humans to identify complex patterns denoting process activities in raw sensor data generated from multiple sources.

In this work, we aim at alleviating the burden of domain experts tasked of identifying process activities from sensor data. We propose a novel semi-automated approach that automatically detects patterns of sequences in sensor data that signify potential occurrences of process activities. Our approach leverages data quantization, sequence abstraction, and sequence segmentation, which assume the existence of repeated subsequences in a sequence of symbols. We focus on processes with highly repetitive activities (e.g., in smart manufacturing), which are particularly suitable for the application of a semi-automated approach, since they are composed of activities performed by machines in a consistent manner over time. The proposed semi-automated approach marks a step forward toward the unsupervised detection of process activities from sensor data in Industry 4.0. As such, it advances the end-user support and increases the quality of process activity detection through automated suggestions. As the approach is validated using real sensor data from processes executed in a small-scale smart factory, its applicability in real Industry 4.0 settings is demonstrated.

This paper is structured as follows: Section 2 introduces background; Section 3 discusses related work; Section 4 introduces our semi-automated approach; Section 5 evaluates the approach; Section 6 concludes the paper.

2. Background and preliminaries

2.1. Activity detection in manufacturing processes

The goal of this work is to detect the execution of *activities* representing units of work that are part of a *business process* (BP). The recent advancements in IoT development have enabled smart devices and machines as performers of activities [16]. However, due to the absence of BPM systems orchestrating or monitoring the activity executions in real-world settings, one has to rely on low-level sensor data emitted from the devices, which first has to be abstracted to a BP-level, to inform about the activity executions [30]. We present this work based on running examples implemented in a small-scale smart factory [27], which features 7 stations to simulate the production of work pieces from raw materials to the final product (cf. Fig. 1). It includes a warehouse (HBW) for storage, an oven (OV) and a milling machine (MM), a sorting machine (SM), a delivery and pickup station (DPS), and a central robot (VGR) for transportation. Each production station consists of various components including sensors (e.g., light barriers, switches, color sensors) and actuators (e.g., motors, compressors, valves) [27]. We consider the data streams emitted from these components as *low-level sensor data* represented as time series, which we aim to abstract to the level of *process activities*. Activities from two discrete manufacturing processes in this factory serve as running examples. The *Storage* process includes the retrieval of new material from DPS by VGR, determining its color, and storing it in HBW. The *Production* process starts with a new order, followed by retrieving an appropriate workpiece from HBW, transporting it to the oven and milling machine to be processed, sorting it in SM, and lastly delivering the product to DPS.

2.2. Repeating substrings and extendibility

Our proposed approach for activity detection relies on the detection of repeating substrings. A *repeating substring* (repeat), refers to a string that occurs at least twice in another string. We denote a repeat as $(l; p_1, p_2, \dots, p_n)$, where l is the length of the repeat, and $p_1, p_2, \dots, p_n$ the set of positions at which the repeat occurs in a given string. Consider the example string $T = \text{abaababa}$ from [24]. It can be checked that $r_1 = (2; 0, 3, 5)$ corresponds to the repeating substring *ab*, which occurs in the positions 0, 3 and 5 of T . Similarly, $r_2 = (3; 0, 3, 5)$ represents the repeated substring *aba*, which occurs in the same positions as r_1 , despite the substrings overlap. We say that r_2 *extends* r_1 adding a symbol

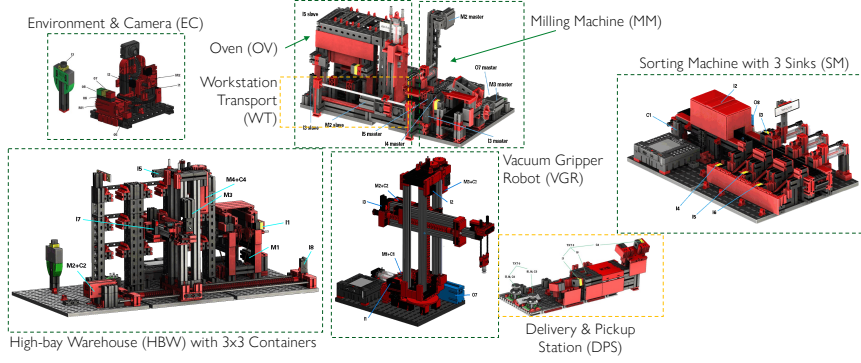


Fig. 1: Production stations of the smart factory used in our work, adapted from [27].

Table 1: Suffix array and other auxiliary arrays for abaababa\$

i	suffixes	sorted suffixes	SA	LCP	BWT
0	abaababa\$	\$	8	0	a
1	baababa\$	a\$	7	0	b
2	aababa\$	aaababa\$	2	1	b
3	ababa\$	aba\$	5	1	b
4	baba\$	abaababa\$	0	3	\$
5	aba\$	ababa\$	3	3	a
6	ba\$	ba\$	6	0	a
7	a\$	baababa\$	1	2	a
8	\$	baba\$	4	2	a

to the right side, thus r_1 is right extendible (RE). In turn, $r_1 = (2; 0, 3, 5)$ is not left extendible (NLE), because all the characters to the left side of the substring **ab** are different (or nonexistent). A repeat is said to be *complete* iff it reports all the occurrences of a substring in the enclosing string. Finally, we say that a repeat is *nonextendible* (NE) iff it is not left extendible nor right extendible. The repeat $r_2 = (3; 0, 3, 5)$ is NE in the example.

2.3. Suffix array and longest common prefix

Our approach also relies on the application of *suffix arrays* (SA), which are data structures used for implementing full-text search indexes [20]. A suffix array is an array of integers storing the positions of all the suffixes of a string, once they have been sorted. Table 1 presents the SA of string $T = \text{abaababa}\$$. The second and third column enumerate the suffixes of T in the original order and lexicographically sorted. By cross-referencing the index i in the first column and the SA in the fourth column, we can check that $\$$ is lexicographically the smallest suffix in T , occurring at position 8, i.e., $T[\text{SA}[0]] = T[8] = \$$. Similarly, the second smallest suffix is $T[\text{SA}[1]] = T[7] = \text{a}\$$, etc. The Burrows-Wheeler transform (BWT) is a reversible permutation on strings. $\text{BWT}[i]$ contains the element preceding the suffix at the i th position in SA. The BWT of our running example is $\text{abbb}\$aaaa$ (cf. last column of Table 1).

All suffixes being lexicographically sorted helps in discovering all the repeats present in string T . In [20], Manber and Myers introduced the LCP array, which stores the length of the longest common prefix substring shared by two adjacent suffixes. For instance, the suffixes in the interval of positions $[3..5]$ have in common the prefix **aba**, as highlighted in Table 1. $\text{LCP}[4] = 3$ states that the suffix $T[\text{SA}[4]]$ and its preceding adjacent suffix $T[\text{SA}[3]]$, both share the prefix **aba**, which is 3 characters long. The information in the LCP array concerns only two suffixes. We define the *LCP value* ℓ of an interval $[i..j]$ with $i, j \in [0..|T|) : i \leq j$, denoted $\ell[i..j]$, as the minimum value in the LCP array associated with all the suffixes in the interval $[i..j]$, i.e., $\ell = \min_{k \in [i+1..j]} (\text{LCP}[k])$. Relying on the containment relation on intervals, all the LCP interval values of a string T can be organized as a rooted tree, the LCP-interval tree. There is a direct connection between the notion of repeats and LCP-interval values, as all LCP-intervals are non-right extendible (NRE) beyond the corresponding LCP-interval value ℓ . However, an LCP-interval can still be left extendible. Non-left extendibility can be checked based on the algorithms in [24, 1]. A repeat can be obtained from an LCP-interval $\ell[i..j]$ by computing $(\ell; \{\text{SA}[k] \mid k \in [i..j]\})$. Linear-time algorithms exist to compute the SA (e.g., [23]), the LCP array (e.g., [18]), the set of LCP-intervals (e.g., [1]), and the BWT can be computed from the SA.

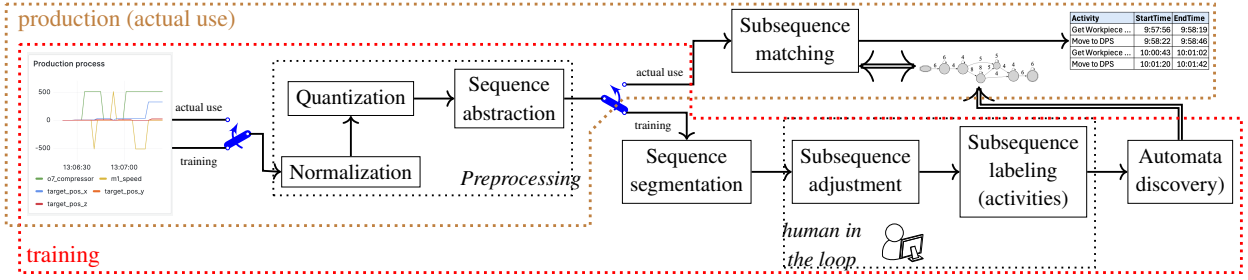


Fig. 2: Overall steps in our approach with training and production phases highlighted

3. Related work

The IoT-meets-BPM manifesto [16] is a major promoter of applying IoT technologies in BPM. It highlights several challenges and opportunities, among which is the IoT-based detection of process activities by leveraging sensors and actuators as sources of real-time data. Since these data are low-level, i.e., they contain no process semantics, an *event abstraction* step is necessary to derive high-level data with process semantics describing activity executions [21]. With collecting data to monitor, visualize and track the progress of a specific task, it is important to notice how some sensors can be involved in different activities of a process. A study that investigates the challenges for automatic connection and derivation of BP data from IoT data [21] highlights some of the main obstacles. These include incomplete descriptions and different meanings of sensor data, resulting in new challenges where sensors might be relevant for scenarios or they have or no significance to the context. Another important challenge is that the analysis of the data will often require human involvement and visualization, e.g., to specify the correct granularity of detected process activities [6].

Approaches investigating IoT-based activity detection have been proposed in [17, 25] and in [28] in terms of *activity detection services*. These services integrate within the general method described in [27], which aims at detecting activities from IoT data sources. The method addresses the challenge of event abstraction, transforming low-level data into high-level data with process semantics, and is applied in the context of smart manufacturing. However, the approach relies on the domain expert to manually label the data, which results in a supervised learning method. Other more specialized approaches have been adopted in the context of smart homes [11, 9] and industrial processes [22]. An underlying challenge common to these approaches lies in bridging the abstraction gap between low-level IoT events and high-level process events [16, 10, 6]. This step is known to potentially introduce ambiguity [13]: leveraging further information such as contextual information is useful to mitigate such a risk [8]. Another fully supervised technique that has shown good results for activity detection—specifically, for human activities—applies the Viterbi algorithm on Hidden Markov Models and Conditional Random Fields [14]. It analyzes long-term activities from sensor data of wearable devices. The authors performed a process including vector quantization, where the data is decomposed into a sequence of primitives where the string of symbols represents an activity and each symbol represents a primitive.

4. Method

On overview of our proposed method is shown in Fig. 2. It begins with a Preprocessing phase including Normalization, (Vector) Quantization, and Sequence Abstraction. It is required in training (red boxes) and in production, i.e., use of the activity detection (brown boxes). Switches show how to select one mode or the other. In training, the next steps are Sequence Segmentation, Subsequence Adjustment and Subsequence Labeling, which require input from the domain expert. Then an automaton is derived from the set of labeled subsequences, which is used in production for matching subsequences from the quantized sequence streamed from the actual process to produce an event log.

4.1. Preprocessing

Normalization and vector quantization. With Vector quantization we reduce the dimensionality of the sensor readings (time-series) by translating them into a sequence of symbols, which are drawn from a finite set of symbols. Here we use

Index	time	m1_sp	m2_sp	m3_sp	cpos_x	cpos_y	cpos_z	tpos_x	tpos_y	tpos_z
1880	10:01:16.740	0	0	0	21.01	0.00	0.00	21.31	0.00	0.00
1884	10:01:17.190	512	0	0	19.51	0.00	0.00	0.00	0.00	0.00
1888	10:01:17.690	512	0	0	9.00	0.00	0.00	0.00	0.00	0.00
1892	10:01:18.200	-512	0	0	0.75	0.00	0.00	0.00	0.00	0.00
1896	10:01:18.630	-512	0	0	10.21	0.00	0.00	0.00	0.00	0.00
1900	10:01:19.060	-512	0	0	19.21	0.00	0.00	0.00	0.00	0.00
1904	10:01:19.490	0	0	-512	21.01	0.00	4.05	21.31	0.00	0.00
1908	10:01:19.920	0	0	-512	21.01	0.00	13.51	21.31	0.00	0.00
1912	10:01:20.350	0	-512	0	21.01	3.75	15.76	21.31	0.00	15.91
1916	10:01:20.780	0	-512	0	21.01	15.01	15.76	21.31	0.00	15.91
1920	10:01:21.200	0	-512	0	21.01	25.06	15.76	21.31	0.00	15.91

(a) Exemplary sensor readings from the vacuum gripper robot (motor speeds, current positions, target positions)

[(b,65), (h,2), (d,2), (g,4), (e,42), (k,56), (i,13), (a,7), (i,1), (k,37), (e,41), (g,8), (d,9), (b,3), (h,9), (d,2), (g,8), (e,35), (k,47), (i,92), (a,14), (c,548), (f,8), (g,13), (d,62), (b,407), (h,2), (d,2), (g,3), (e,43), (k,54), (i,14), (a,9), (i,1), (k,36), ...]

(b) Run-length encoded sequence of quantized values

Fig. 3: Dataset excerpt (a) and quantized sequence (b)

clustering to map sensor readings to a cluster label. Before clustering, the sensor data must be normalized. Here, we consider the min-max scaler, i.e., for each sensor, the minimum and the maximum values in the dataset are determined and then each sensor value is divided by the difference between the minimum and maximum to get a new value in the interval [0, 1]. The normalized data from multiple sensors is aggregated to form vectors. More specifically, sensors are aggregated for each station in the smart factory (cf. Section 2.1). These vectors are the input for clustering using *k-means* [15]. Fig. 3a presents an excerpt of the sequence of sensor readings from our dataset taken from the VGR station [26]. The sensor readings are normalized and a vector is aggregated per row, which is then used to train a *k-means* clustering model. The same normalization and clustering models are used during training and in production to generate a sequence of uniformly quantized values, which corresponds to the sequence of cluster labels. In cases where the number of clusters is greater than 10, English letters are used as cluster labels for readability.

Sequence abstraction. The sequences of quantized values includes subsequences repeating the same symbol. Fig. 3b presents a sequence of quantized values, encoded according to the run-length scheme, i.e., each tuple specifies the character and the number of its adjacent occurrences (e.g., the tuple (b,65) indicates that the symbol b is repeated 65 times in a row). However, the frequency with which a single character is observed to be duplicated is not uniform. In our setting, such variability is a result on the choice of sampling rate and latencies in the physical operations. Here, we decided to replace repeated symbols with a single occurrence of the symbol, called “Sequence abstraction”. Fig. 4 shows the abstracted deduplicated sequence of our running example.

4.2. Sequence segmentation

Our approach is based on the identification of repeating subsequences in the sequence of quantized values. The approach uses the parameters minimum length (min-length) and minimum support (min-support) to reduce the number of NE repeats to be considered for further analysis. These parameters are adjusted by considering the characteristics of the dataset. We expect the dataset to record the execution of multiples instances of the underlying process, i.e., each activity is repeated multiple times, so that we can define a min-support as a parameter in the training phase. Moreover, we observed that subsequences of short length are very frequent, hence we use a parameter min-length to narrow the number of candidates (e.g., 3 or more symbols). Note that min-support and min-length are parameters commonly used in substring mining. We also consider that re-initialization phases might manifest themselves as non-trivial sequences, e.g., the robot arm returning to a resting position. The sequence in Fig. 4 shows how our assumptions hold in context. The highlighted subsequence *gdbhg* is repeated 19 times in the quantized example sequence. The subsequence *gdbhg* is a non-extendible repeated substring: it is preceded by a letter in {e, f, j} and followed by a letter in {e, f}. We proceed by computing the set of NE repeated substrings in the sequence as described in Section 2.3. The approach relies on the parameters *min-length* and *min-support* to reduce the number of NE repeats to be considered for further analysis. Fig. 5 presents the NE repeats of the example, considering a min-length of 4 symbols and a min-support of 5. We present only the top 10 as the number of NE repeated strings could be large and overwhelming for the domain expert who would analyze the subsequences taking their support into account.

[illegible]

Fig. 4: Deduplicated sequence of quantized values (sequence abstraction) with a frequent NE repeat in context

bhdg	20	gdbhdgekia	12
gdbhdg	19	kegd	10
bhdgek	15	kegdgbhdg	9
gdbhdge	14	bhdgekiacfgdbhdg	5
bhdgekia	13	gdbhdgfgdgfcajgdbhdgek	5

Fig. 5: Top 10 non-extendible repeats with support (# appearances) for the running example



Fig. 6: Screenshot of the visualization tool (1: top 10 repeated substrings; 2: simplified selected sequence; 3: quantized values)

4.3. Subsequence adjustment and activity labeling (human-in-the-loop)

In the previous step, we determined the set of repeating subsequences with higher support that may provide indications of activity executions to the user. However, presenting the quantized, abstracted subsequence as suggestions for identified activities is not meaningful for the domain expert. Therefore, we run a decision tree learning [15] over the input dataset using the cluster labels as target classes. Then, we retrieve from the decision tree the predicates that lead to each one of the leaf nodes, which are associated to a single cluster label. Such predicates (disjunctions of conjunctions if more than one leaf is associated with a cluster label) are then presented to the domain expert as they are more understandable. To illustrate the idea, consider the screenshot of our prototype in 6. The sequence of quantized values without further processing (i.e., abstraction) is displayed in area 3. The top 10 repeated substrings can be selected by the domain expert in the area 1. In the screenshot, the first subsequence is selected. In area 2 a simplified view of the sequence is presented. All cluster labels have the same duration to help the domain expert in the analysis. The user can select any cluster of the subsequence, and the set of predicates (conjunction) for that cluster is shown in the box on the right side of area 2. The domain expert can assign an activity label to a subsequence in the text field just below area 1, from which the automaton for detecting this activity in production is then derived (cf. Fig. 2).

In general, the subsequence with higher support does not always correspond to a re-initialization phase. However, in the factory setup this is reasonable because a re-initialization phase is associated with the VGR going back to its resting position, which occurs always between the activities that involved the robotic arm. Based on this observation, we added the possibility of discarding a subsequence, which will force a segmentation over the input quantized sequence. The resulting subsequences with the associated activity labels provided by the domain expert are shown in Table 2. This set of subsequences is used to derive a deterministic automaton for each activity. Using standard techniques from

Table 2: Precision and recall of the activity detection (incl. activity labels for subsequences), with and without adjustments by domain expert

Provided Activity Label (Subsequence)	w/o adjustments		Allen's relation	with adjustments		# discarded symbols
	Prec	Rec		Prec	Rec	
Move to DPS (ekiacfgd)	0	0	Overlaps	1	1	3 (right)
Get Workpiece from DPS (ekiaikegd)	1	1	Contains	1	1	0
Pickup and transport to Oven (fgdgfcajgd)	0	0	Contained by	1	1	1 (left)
Pickup and transport to Sink (ekaiakegd, ekiakegd)	0	0	Contained by	1	1	1 (left)

automata theory, the automata can be merged to derive a single automaton that recognizes all activities [2]. With this automaton, it is possible to detect activities from a stream of sensor readings either online or offline.

5. Evaluation and discussion

All the concepts described before were implemented in a prototype using the Python programming language and well-known data analytics and visualization libraries (e.g., Pandas, sklearn, Plotly). Fig. 6 presents a screenshot taken from the implementation of the prototype. To evaluate the usefulness of our method and prototype, we conducted an experiment using a dataset generated with the smart factory and processes described in Section 2.1. The dataset is available at [26]. The training dataset was produced by executing the production and storage processes 5 times each, sampling the sensors at a rate of 10 Hertz. The proposed approach was able to identify from the test dataset, with the aid of the domain expert, 4 activities from the set of discovered subsequences. The processes were controlled using BPMN 2.0-based process models, which were executed by the Camunda BPM system [29]. The dataset includes an event log generated by the BPM system, which represents a ground truth to compare with and to assess the accuracy of the activities detected by our prototype. More details on the dataset can be found in [26].

Given that an activity execution is defined by an interval in time from its start to end, we leverage Allen's interval relationships [3] to assess the accuracy with which an activity was detected w.r.t. the ground truth. The work in [19] supports several accuracy metrics that are appropriate here. We assume that the boundaries at the beginning and/or end of a repeating subsequence are ultimately adjusted by the domain expert to align with the ground truth. Hence, we consider that only the relationships shown in Table 2 are relevant in our context. We computed precision and recall of the activities detected by our prototype with adjustments by the domain expert on the datasets as shown in Table 2. Precision and recall have high values, demonstrating that our approach is a suitable solution to support the domain expert. The method was able to automatically discover the subsequence associated with activity "Get Workpiece from DPS". All other activities required the input of the expert. A drawback of the approach is that it is sensitive to variation in the sensor data, because it uses automata to implement the subsequence matching over a stream of sensor readings. This limits its applicability in domains with higher human involvement in activity executions. Note that state-of-the-art algorithms for repeating subsequence discovery have linear time and space complexities, making them capable of handling large datasets efficiently.

6. Conclusions

We propose a semi-automated approach to detecting process activities from sensor data as an event abstraction step based on frequently repeated subsequences and automata. We evaluated the approach with a proof-of-concept implementation and real data from a small-scale Industry 4.0 smart factory. In contrast to the fully manual labeling of sensor data, the domain expert mainly checks the suggested subsequences and adjusts them if required. The approach is based on an automatic discovery of a set of non-exendible repeated subsequences and relies on the interaction with a domain expert to improve the accuracy of the method. In the future, our aim is to extend our technique by using an error-correcting automaton to be less sensitive to variations in sensor data. Another path we are exploring is the use of online conformance checking techniques [7, 12]. We plan to further evaluate our prototype with other datasets and perform a comparative analysis with existing approaches (e.g., [14]). Finally, we also consider running empirical evaluations with the participation of domain experts in real-world processes.

Acknowledgements

This work has received funding from the Swiss National Science Foundation under Grant No. IZSTZ0_208497.

References

- [1] Abouelhoda, M.I., Kurtz, S., Ohlebusch, E., 2004. Replacing suffix trees with enhanced suffix arrays. *J. Discrete Algorithms* 2, 53–86.
- [2] Aho, A.V., Sethi, R., Ullman, J.D., 1986. *Compilers: Principles, Techniques, and Tools*. Addison-Wesley.
- [3] Allen, J.F., 1983. Maintaining Knowledge about Temporal Intervals. *Commun. ACM* 26, 832–843.
- [4] Babangida, L., Perumal, T., Mustapha, N., Yaakob, R., 2022. Internet of things (iot) based activity recognition strategies in smart homes: a review. *IEEE sensors journal* 22, 8327–8336.
- [5] Baheti, R., Gill, H., 2011. Cyber-physical systems. *The impact of control technology* 12, 161–166.
- [6] Beerepoot, I., Di Ciccio, C., Reijers, H.A., Rinderle-Ma, S., Bandara, W., Burattin, A., Calvanese, D., Chen, T., Cohen, I., et al., 2023. The biggest business process management problems to solve before we die. *Comput. Ind.* 146, 103837.
- [7] Burattin, A., Carmona, J., 2017. A framework for online conformance checking, in: *Business Process Management Workshops - BPM 2017 International Workshops, Revised Papers*, Springer. pp. 165–177.
- [8] Di Federico, G., Burattin, A., 2023. Cvamos—event abstraction using contextual information. *Future Internet* 15, 113.
- [9] Di Federico, G., Nikolajsen, E.R., Azam, M., Burattin, A., 2021. Linac: A smart environment simulator of human activities, in: *International Conference on Process Mining*, Springer International Publishing Cham. pp. 60–72.
- [10] Diba, K., Batoulis, K., Weidlich, M., Weske, M., 2020. Extraction, correlation, and abstraction of event data for process mining. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 10, e1346.
- [11] Esposito, L., Leotta, F., Mecella, M., Veneruso, S., 2022. Unsupervised segmentation of smart home logs for human habit discovery, in: *2022 18th International Conference on Intelligent Environments (IE)*, IEEE. pp. 1–8.
- [12] Franceschetti, M., Seiger, R., González, M.J.G., García-Ceja, E., Flores, L.A.R., García-Bañuelos, L., Weber, B., 2023a. Proambition: Online process conformance checking with ambiguities driven by the internet of things., in: *CAiSE Research Projects Exhibition*, pp. 52–59.
- [13] Franceschetti, M., Seiger, R., López, H.A., Burattin, A., García-Bañuelos, L., Weber, B., 2023b. A Characterisation of Ambiguity in BPM, in: *Proc. of ER 2023*, Springer. pp. 277–295.
- [14] García-Ceja, E., Brena, R.F., Carrasco-Jiménez, J.C., Garrido, L., 2014. Long-Term Activity Recognition from Wristwatch Accelerometer Data. *Sensors* 14, 22500–22524.
- [15] Han, J., Kamber, M., 2006. *Data Mining: Concepts and Techniques*, Second Edition. Elsevier.
- [16] Janiesch, C., Koschmider, A., Mecella, M., Weber, B., Burattin, A., Di Ciccio, C., Fortino, G., et al., 2020. The Internet of Things meets Business Process Management: A manifesto. *IEEE Trans. Syst. Man. Cybern. Magazine* 6, 34–44.
- [17] Janssen, D., Mannhardt, F., Koschmider, A., van Zelst, S.J., 2020. Process model discovery from sensor event data, in: *International Conference on Process Mining*, Springer. pp. 69–81.
- [18] Kasai, T., Lee, G., Arimura, H., Arikawa, S., Park, K., 2001. Linear-Time Longest-Common-Prefix Computation in Suffix Arrays and Its Applications, in: *Proc. of CPM’01*, Springer. pp. 181–192.
- [19] Kurz, A.F., Seiger, R., Franceschetti, M., Weber, B., 2024. Activity and sequence detection evaluation metrics: A comprehensive tool for event log comparison, in: *Proc. Best Dissertation Award, Doctoral Consortium, and Demonstration & Resources Forum at BPM 2024*, pp. 76–80.
- [20] Manber, U., Myers, G., 1990. Suffix Arrays: A New Method for On-Line String Searches, in: *Proc. of ACM-SIAM’90*, SIAM. pp. 319–327.
- [21] Mangler, J., Seiger, R., Benzin, J.V., Grüger, J., Kirikkayis, Y., et al., 2024. From Internet of Things Data to Business Processes: Challenges and a Framework. [arXiv:2405.08528](https://arxiv.org/abs/2405.08528).
- [22] Mannhardt, F., Bovo, R., Oliveira, M.F., Julier, S., 2018. A taxonomy for combining activity recognition and process discovery in industrial environments, in: *International Conference on Intelligent Data Engineering and Automated Learning*, Springer. pp. 84–93.
- [23] Nong, G., Zhang, S., Chan, W.H., 2011. Two Efficient Algorithms for Linear Time Suffix Array Construction. *IEEE Trans. Computers* 60, 1471–1484.
- [24] Puglisi, S.J., Smyth, W.F., Yusufu, M., 2010. Fast, Practical Algorithms for Computing All the Repeats in a String. *Math. Comput. Sci.* 3, 373–389.
- [25] Rebmann, A., Emrich, A., Fettke, P., 2019. Enabling the discovery of manual processes using a multi-modal activity recognition approach, in: *Business Process Management Workshops: BPM 2019 International Workshops, Vienna, Austria, September 1–6, 2019, Revised Selected Papers* 17, Springer. pp. 130–141.
- [26] Seiger, R., 2024. Dataset from a smart factory to evaluate a semi- automated approach to detecting process-level activities from sensor data. doi:[10.5281/zenodo.14441997](https://doi.org/10.5281/zenodo.14441997).
- [27] Seiger, R., Franceschetti, M., Weber, B., 2023a. An Interactive Method for Detection of Process Activity Executions from IoT Data. *Future Internet* 15, 77.
- [28] Seiger, R., Franceschetti, M., Weber, B., 2023b. Data-Driven Generation of Services for IoT-Based Online Activity Detection, in: *Proc. of ICSOC 2023*, part 2, Springer. pp. 186–194.
- [29] Seiger, R., Malburg, L., Weber, B., Bergmann, R., 2022. Integrating process management and event processing in smart factories: A systems architecture and use cases. *Journal of Manufacturing systems* 63, 575–592.
- [30] Weber, B., Abbad-Andaloussi, A., Franceschetti, M., Seiger, R., Völzer, H., Zerbato, F., 2023. Leveraging digital trace data to investigate and support human-centered work processes, in: *Intern. Conf. on Evaluation of Novel Approaches to Software Engineering*, Springer. pp. 1–23.