Clustering Raw Sensor Data in Process Logs to Detect Data Streams

Matthias Ehrendorfer(✉), Juergen Mangler, and Stefanie Rinderle-Ma

Technical University of Munich, TUM School of Computation, Information and Technology, Garching, Germany
{matthias.ehrendorfer,juergen.mangler,stefanie.rinderle-ma}@tum.de

Abstract. The execution and analysis of processes is strongly influenced by sensor streams, e.g., temperature, that are measured in parallel to the process execution and stored in process event logs. This holds particularly true for application domains such as logistics and manufacturing. However, currently, these sensor streams are collected and stored in an arbitrary and unsystematic way. Hence, this work proposes an approach that prepares sensor streams into individual data streams that can be annotated to process tasks and used for process analysis and prediction.

Keywords: Process Models · Process Model Enhancement · Annotation of Context Data · Process Logs

1 Introduction

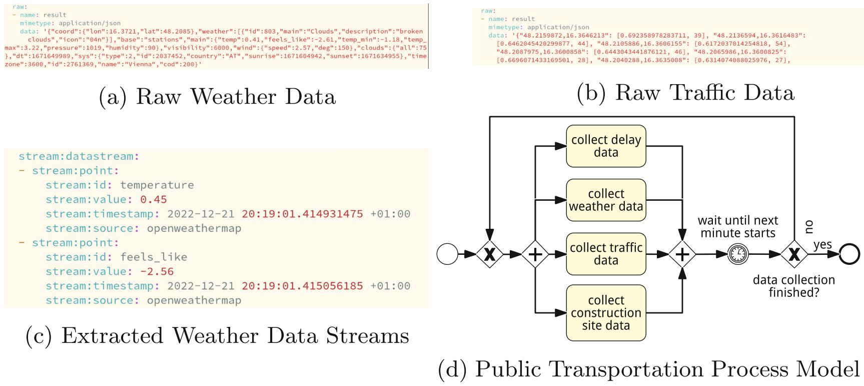
Data is paramount to drive and optimize process execution, i.e., at decision points in the process model and as input/output for services, application programs, and human actors invoked by the process tasks [1,13]. In addition to this *intrinsic data*, *extrinsic data* might affect the process execution, as well, for example the process outcome [4] or the prediction of concept drift [15]. Extrinsic data comprises raw data available in a machine participating in the process, or sensor data monitoring the environment in which the process is enacted. Recently, the DataStream XES extension (cf. [11]) has been proposed in order to enable the recording of sensor streams in process event logs.

Consider the realistic transportation scenario [10] depicted in Fig. 1. The process model shown in Fig. 1d collects multiple measurements relevant to an underlying public transport process, i.e., delay, weather, traffic, and construction sites, as response of one service call. The resulting data is logged in the XES SensorStream format. The raw sensor streams for weather and traffic are depicted in Fig. 1a and Fig. 1b respectively. As can be seen from the weather sensor stream,

This work has been partly funded by the Austrian Research Promotion Agency (FFG) via the "Austrian Competence Center for Digital Production" (CDP) under the contract number 881843. This work has been supported by the Pilot Factory Industry 4.0, Seestadtstrasse 27, Vienna, Austria.

M. Sellami et al. (Eds.): CoopIS 2023, LNCS 14353, pp. 438–447, 2024.
https://doi.org/10.1007/978-3-031-46846-9_25

multiple measurements are contained, e.g., temperature, wind, or pressure, in an arbitrary and hence unsystematic way. In order to utilize the sensor streams for process analysis and predictive process monitoring, the sensor streams are to be prepared, i.e., relevant sensor information is to be extracted from the raw stream and clustered into individual data streams. These data streams can then be annotated to process tasks such that, in the sequel, the data streams can be already collected in a systematic way.



```
raw:
- name: result
  mimetype: application/json
  data: '{"coord":{"lon":16.3721,"lat":48.2085},"weather":[{"id":803,"main":"Clouds","description":"broken
    clouds","icon":"04n"}],"base":"stations","main":{"temp":0.41,"feels_like":-2.61,"temp_min":-1.18,"temp_
max":3.22,"pressure":1019,"humidity":90},"visibility":6000,"wind":{"speed":2.57,"deg":150},"clouds":{"all":75
},"dt":1671649989,"sys":{"type":2,"id":2037452,"country":"AT","sunrise":1671604942,"sunset":1671634955},"time
zone":3600,"id":2761369,"name":"Vienna","cod":200}'
```

(a) Raw Weather Data

```
raw:
- name: result
  mimetype: application/json
  data: '{"48.2159872,16.3646213": [0.692358978283711, 39], "48.2136594,16.3616483":
    [0.6462045420299877, 44], "48.2105886,16.3606155": [0.6172037014254818, 54],
    "48.2087975,16.3600858": [0.6443043441876121, 46], "48.2065986,16.3600825":
    [0.6696071433169501, 28], "48.2040288,16.3635008": [0.6314074088025976, 27],
```

(b) Raw Traffic Data

```
stream:datastream:
- stream:point:
    stream:id: temperature
    stream:value: 0.45
    stream:timestamp: 2022-12-21 20:19:01.414931475 +01:00
    stream:source: openweathermap
- stream:point:
    stream:id: feels_like
    stream:value: -2.56
    stream:timestamp: 2022-12-21 20:19:01.415056185 +01:00
    stream:source: openweathermap
```

(c) Extracted Weather Data Streams

collect delay data
collect weather data
collect traffic data
collect construction site data
wait until next minute starts
no
yes
data collection finished?

(d) Public Transportation Process Model

Fig. 1. Public Transportation - External Service Log Data and Process Model

Explicitly annotating information about how and which data is collected in individual tasks of a process model is necessary for "Placing Sensors in a Process-Aware Way" [6]. However, as doing so manually is time-consuming, cumbersome, and error-prone, this paper provides a sensor stream extraction and fusion approach that constitutes the prerequisite for future task annotation. The approach (1) breaks down raw sensor streams in process event logs into comparable components, (2) describes how to determine a distance between these components in order to enable clustering and (3) explores different methods of clustering the collected context data to find individual data streams (sensor stream fusion).

The approach is evaluated using a synthetically created data set which portrays weather data and is used to demonstrate the applicability of the approach and a real-world data set from the manufacturing domain which contains context data from the machine tool and measuring machine used in the process.

Section 2 describes the approach presented in this paper, Sect. 3 contains the evaluation of the approach and Sect. 4 discusses the results. Furthermore, Sect. 5 gives an overview over related work and the paper is concluded in Sect. 6.

2 Context Data Clustering Approach

As motivated in the introduction and the transportation use case (cf. Fig. 1), sensor data streams collected as context data during process execution, currently, cannot be directly processed for process analysis and prediction due to the following reasons:

1. Sensor data might occur at "random" times from the point of view of the process as machines and sensors might not always send the same data or external endpoints are not under the control of the process.
2. Endpoints might provide different data depending on their implementation or be changed over time leading to inhomogeneous sensor data.
3. Sensor streams might contain multiple measurements, e.g., different data streams of a machine or different sensor readings are combined.
4. Due to the inhomogeneity, the raw sensor data does not have any schema.
5. It is unclear which sensor streams or parts of sensor streams are connected to the process instance or to single process tasks.

The proposed approach aims at tackling 1.–4. by breaking down the raw sensor streams into comparable components and then based on a structural (cf. Sect. 2.1) and value-based (cf. Sect. 2.2) clustering as well as based on a combination of both (cf. Sect. 2.3), fusing these components into individual, homogeneous data streams that can be connected to tasks and build the basis for process analysis and prediction.

2.1 Structural Analysis

The goal of the structural analysis is to find components of the raw sensor streams which are similar regarding their structure, i.e., they provide a value/timestamp pair with a certain label, they contain the same types of measurements (e.g., numerical temperature reading or textual description of the noise level), or any other structural similarity. Structural similarity is calculated using the JSON edit distance (JEDI) [5] that quantifies how similar two JSON documents are considering their structure. More precisely, the JEDI distance is calculated based on the number of edit operations (add, delete, rename) necessary to transform one structure into the other.

2.2 Value-Based Analysis

Even when data is structurally similar, it might still belong to different data streams based on its values (e.g., different measuring units are used or measurements are taken at completely different times). Calculating the distance between two sensor stream components regarding their values is not straightforward as multiple types of data values might occur. We compare the values of two sensor stream components as follows: Each value of the first component is compared to all values of the other component. Depending on the type of the values we use (1) Levenshtein distance [9] for strings, (2) time period between two values for timestamps, and (3) difference for numbers. The result is a mxn matrix of distances between all data values. For each value the lowest distance to the other component is then added to the overall distance for this type of value. As a result a distance from one component to another is generated for each value type.

Distances of different value types might not be comparable to each other. Hence, they are scaled by dividing the value calculated in the first place by the maximum distance between all context data components for the respective value type. The results for each data type are then combined - weights can be chosen based on the scenario. Other types of values and other distance measures for the presented data types (string, timestamp, number) can be added easily.

2.3 Combining Structural and Value-Based Analysis

This section describes the steps of the overall approach based on the two analysis methods described in Sects. 2.1 and 2.2.

Step 0 - Extract Raw Sensor Stream Data From Event Logs: The extraction results in a list of sensor data elements collected at different points in time and by different events.

Step 1 - Break Down Sensor Stream Data Into Components: The extracted data is broken down into its components by using the whole raw data as starting point and then recursively adding available children (e.g., sensor measurement consists of temperature and humidity) to the components. This allows to compare different components of the sensor data as for some scenarios bigger parts of the original raw data are comparable while for other scenarios only lower level components (e.g., single value/timestamp pairs) can be compared.

Step 2 - Choose Strategy: When using **Strategy A** first structural analysis (cf. Sect. 2.1) is performed and afterwards the clusters are refined using value-based analysis (cf. Sect. 2.2). If **Strategy B** is used the order is reversed: first value-based analysis (cf. Sect. 2.1) is used and then clusters are refined using structural analysis (cf. Sect. 2.2). **Strategy C** only uses one kind of analysis, i.e., represents Strategy A or B but stopping after Step 3.

Strategy A: Initially use Structural Analysis and Afterwards Refine Using Value-Based Analysis:

- **Step 3A - Cluster Components Based on Structural Analysis:** The distance between components is found using structural analysis (cf. Sect. 2.1) and then used for clustering. We opt for using DBSCAN for clustering as the number of clusters does not have to be defined. We will experiment with other clustering approaches such as k-means in the future.
- **Step 4A - Refine Individual Clusters Based on Value-Based Analysis:** Afterwards, (value-based) distances (cf. Sect. 2.2) between components within structural clusters are used to build refined clusters (again using DBSCAN).

Strategy B: Initially use Value-Based Analysis and Afterwards Refine Using Structural Analysis:

- **Step 3B - Cluster Components Based on Value-Based Analysis:** The distance between components is found using value-based analysis (cf. Sect. 2.2) and then used for clustering (using DBSCAN).

- **Step 4B - Refine Individual Clusters Based on Structural Analysis:** Afterwards, (structural) distances (cf. Sect. 2.1) between components within value-based clusters are used to refine clusters (again using DBSCAN).

Strategy C: Only Consider Structural or Value-Based Analysis
This strategy considers either structural (C1) or value-based (C2) aspects of the components. Therefore, it is a modification of Strategy A (for C1) or B (for C2) where refinement is skipped (i.e., steps 4A and 4B are omitted).

3 Evaluation

The evaluation is performed on an artificial data set as well as on a real-world data set from the manufacturing domain. Code, data and instructions on how to run the code are available at gitlab[1].

Methodology: For both data sets, we first apply Strategy C in both variants, i.e., C1 based on structure and C2 based on values. C1 is then refined into Strategy A, i.e., structure-based clusters are refined into value-based ones, and C2 is refined into Strategy B, i.e., value-based clusters are again clustered based on the structure. The results can be shown in tables (see Table 1): the leftmost column represents the firstly built clusters (in this case structural), the second column represents the (in this case value-based) refinement. A "*" denotes that the original cluster before refinement is described. Entries in the following columns show that data components of this data stream can be found in this cluster. Apart from Table 1 only summarized results are reported by giving information about "Clusters per Data Stream" (CpDS) and "Data Streams per Cluster" (DSpC) for structural (Struct) and value-based (VB) clusters which allow to estimate the effectiveness of a strategy. A perfect result would be one where CpDS and DSpC are 1 for all clusters and data streams because then one cluster represents exactly one data stream and one data stream is represented by exactly one cluster.

For structural clustering an epsilon of 0.1 is used which means that components in a cluster have the exact same structure. A higher epsilon would lead to less similar components being in the same cluster and thus more imprecise results. Weights for the value-based analysis have been set so that all data types are considered with equal weight. The remaining parameters are explained in the relevant sections. For all results only clusters with >1 elements are considered.

3.1 Artificial Data Set

The following sensor measurements being "measured" in two different time slots on subsequent days are included in the artificial data:

- Temperature: **value in degree Celsius** (between −5 and 20), **value in degree Fahrenheit** (between 268 and 295), **short textual description** (e.g., "hot", "cold"), and **long textual description** (e.g., "Today the weather is very hot and it is expected that ...")

[1] https://gitlab.com/me33551/semi_automatic_context_data_extraction.

- Humidity: value providing **relative humidity** (between 40 and 90), **short textual description** (e.g., “high”, “low”), **long textual description** (e.g., “We expect tropical weather with a high humidity for today.”)

Strategy C1 and A: Using only structural analysis (Strategy C1) the results show that the clusters already provide some grouping regarding data streams contained in the data components of a cluster (cf. rows with “*” in column “VB” in Tab. 1 where cluster 2 contains textual data streams and cluster 5 contains all other data streams). Furthermore, some structural clusters (e.g., 3, 4, 6, ...) are already identified as not containing information representing any data stream i.e., they just contain single values or components including data from multiple streams. Looking at the “CpDS Struct” and “DSpC Struct” values it can be seen that each stream is contained in only one cluster (all CpDS Struct values are 1) but the problem is that for a component in a cluster it cannot be clearly decided to which data stream they belong (DSpC Struct values are 8 and 6).

When refining the structural clusters as described in Sect. 2.3 the results reported in Table 1 show that the refined clusters represent nearly all data streams available in the data set. This can be seen because all apart from 3 “CpDS VB” values are greater than 0. Also all but one “DSpC VB” values are 1 (and one is 2). This means that all but one of the refined clusters contain only one data stream. This is a good result because overall it means that all components can be assigned to a data stream based on the cluster in which they are.

Table 1. Artificial Data Set Results for Strategy C1 and A

Clusters		Temperature								Humidity							
Struct	VB	Celsius		Fahren-heit		Short Text		Long Text		Relative		Short Text		Long Text		DSpC Struct	DSpC VB
		Day 1	Day 2	D1	D2	D1	D2	D1	D2	D1	D2	D1	D2	D1	D2		
2	*					x	x	x	x			x	x	x	x	8	
	0							x									1
	2						x						x				2
	3					x											1
	5														x		1
	6											x					1
	7							x									1
	9						x										1
	11														x		1
3	N/A															N/A	
4	N/A															N/A	
5	*	x	x	x	x					x	x					6	
	0			x													1
	1	x															1
	3									x							1
	5										x						1
	6									x							1
	7				x												1
	8										x						1
	9			x													1
6	N/A															N/A	
8	N/A															N/A	
9	N/A															N/A	
12	N/A															N/A	
CpDS Struct		1	1	1	1	1	1	1	1	1	1	1	1	1	1		
CpDS VB		1	0	2	1	1	2	2	0	2	2	1	1	0	2		

Strategy C2 And B: When applying Strategy C2 (using only value-based analysis) only one cluster is found because all components are connected (distance of 0) via the root component. This is because values contained in lower level components (containing one or two values) are also included in higher level components (as well as the root). Therefore, the results shown in the "Artificial" section for Strategy B of Tab. 2 are only based on components which contain exactly two values (i.e., a value/timestamp pair). This results in "CpDS VB - B" values being the same values as for Strategy A. DSpC values are 1 for all but one data stream (where it is 2) meaning that in one of the clusters components from 2 data streams is included (components in other clusters can be easily allocated as their "DSpC VB" is 1 and therefore each cluster represents only one data stream).

Refinement using structural analysis (cf. Sect. 2.3) does not lead to new clusters because the exclusion as described in the paragraph above where only components with two values are used leads to structurally similar components. Even if refinements could be made this would not make sense because the clusters found using only value-based analysis already lead to a nearly perfect result with only one "DSpC VB" value not being 1. However, extensive domain knowledge about the internal structure of collected data is needed in order to select the components when starting with value-based analysis as in Strategy C2 and B while the results presented in Sect. 3.1 (using Strategy A) lead to comparable results without any prior knowledge.

Table 2. Summarized Results for Artificial and Real-World Data Sets

Artificial	Temperature								Humidity					
	Celsius		Fahrenheit		Short Text		Long Text		Relative		Short Text		Long Text	
	Day 1	Day 2	D1	D2	D1	D2	D1	D2	D1	D2	D1	D2	D1	D2
CpDS Struct - A	1	1	1	1	1	1	1	1	1	1	1	1	1	1
CpDS VB - A	1	0	2	1	1	2	2	0	2	2	1	1	0	2
CpDS VB - B	1	0	2	1	1	2	2	0	2	2	1	1	0	2

Real-World	Active Power			keyence	Spindle													
	A	B	C		act Speed	drive Load	aaLeadP			aaTorque			aaVactB			aaLoad		
							X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
CpDS Struct - A	1	1	1	1	1	1	1	0	0	1	1	1	1	0	0	1	1	1
CpDS VB - A	4	4	4	6	2	2	2	0	0	2	2	1	2	0	0	2	2	1
CpDS VB - B	4	4	4	5	1	1	1	0	0	1	1	1	1	0	0	1	1	1

3.2 Real-World Data Set

The real-world data set[2] contains log files from a manufacturing process including data from (1) a robot handling transportation of the part between stations (2) the machine tool producing a part, and (3) measuring data from quality control of a part. Only part of the data available in the data set is used for the evaluation. Due to the high number of context data we focused on 3 different log events within one process instance and used the first 151 components of each event for the analysis. This already includes most of the data streams (i.e., only aaLeadP Y and Z and aaTorque Y and Z are not present in any cluster (see results).

[2] https://cpee.org/~demo/DaSH/batch14.zip [Online; accessed 15-Jul-2023].

Strategy C1 And A: Considering only structural analysis (Strategy C1) the results show that most of the data streams are included in the clusters (only 4 “CpDS Struct - A” values in Table 2 are 0). The “DSpC Struct” values are 4 and 10 meaning that the two structural clusters found contain this number of data streams. Root and high-level components are excluded from this structural analysis because a distance measure based on edit distance on such big data structures is very costly. Furthermore, these components would be in their own clusters because the epsilon with 0.1 allows only structurally equal components in the same cluster.

Refining the results described above (Strategy A) leads to “CpDS VB - A” values between 1 and 6 (apart from the 4 data streams with 0). The “DSpC VB” values are all 1 in one of the original structural clusters and between 2 and 4 in the other one. Therefore, refined clusters with a “DSpC” of 1 only contain components belonging to one data stream while for the ones with higher values it at least restricts the number of data streams to which components in this cluster belong.

Strategy C2 and B: As for the artificial data set (see Sect. 3.1) it is necessary to limit the number of values in the examined components to prevent one big cluster - therefore, only components with a minimum of 2 values and a maximum of 15 values are used. All but 4 of the data streams are found in one of the clusters (“CpDS VB - B” in Table 2 bigger than 0 for all but 4 data streams). The “DSpC VB” values are 1 for all clusters containing components with “keyence” or “Active Power” measurements. However, the other cluster has a “DSpC VB - B” value of 10 which means that it cannot be decided to which of these data streams a component in this cluster belongs. Furthermore, as in Sect. 3.1 refinement for Strategy B is not possible and finding the right parameters for the minimum and maximum number of values again requires in-depth domain knowledge.

4 Discussion

The evaluation shows that detecting data streams based on raw data included in logged events is possible. However, because the approach deconstructs all data received into its components and calculates distances between each of them for clustering this leads to a long calculation time. A run-time version needs to either reduce the amount of data or to not compare every component to each of the others. Another limitation is that some parameters need to be set (depending on the strategy used). This requires knowledge about the domain and collected data. For future work a user interface to inspect different combinations of parameters would be an option. However, for a fully automated approach another solution would be needed. Overall, the presented approach builds clusters representing different data streams collected in a process. This information can be used to create data schemas over all components in a cluster and use them for automatic extraction of data from raw event data loads. However, generating a schema which fits a cluster structurally and value-wise needs to be further investigated.

5 Related Work

Recent process mining papers such as [12] introduce the importance of the data perspective. [16,17] exploit textual information as additional source of unstructured data to improve process analysis results. Other examples include exploiting the sentiment for news data for remaining time prediction in [18] and [15] describing an approach to identify concept drifts based on sensor data. [2] proposes to predict process performance indicators based on identification of relevant context information through domain knowledge and expert feedback. [8] and [14] use sensor data as basis to identify process activities and discover a process model.

Another related area is Complex Event Processing (cf. [3]) where rules for events are defined to filter events and perform analysis. In contrast, our approach tries to find information about data streams in the process from the context data contained within events without prior definition of rules.

Our approach uses JSON edit distance (cf. [5]) which is an adoption of the well-known edit distance for XML documents to calculate the distance between two components. Other works in the context of NoSQL data stores deal with providing schemas for semi-structured JSON data as well as structural similarity measures (cf. [7]) or data handling in more specific cases (e.g., considering hidden data available as meta data or conceptual schema extraction).

6 Conclusion

This paper describes how to identify data streams appearing in a process by analyzing the raw data load contained in logged events. This includes making raw sensor streams comparable by breaking them down into components and calculating a distance between them based on structure or included values. Afterwards, different strategies to find clusters representing data streams occurring in the process are compared and discussed. The evaluation shows that using the presented approach the data stream to which a component belongs can be narrowed down based on its assigned cluster. Furthermore, it is discussed that when value-based analysis is performed without prior structural analysis (i.e., Strategy C2 and B) some components have to be excluded to still achieve meaningful results. However, this filtering requires domain knowledge which is not needed for Strategy C1 and A. Future work will further investigate how the components contained in a cluster can be used to create a schema for the data stream represented by it so that this information can be used to annotate data streams to process tasks to be used for process analysis and prediction.

References

1. Brunk, J., Stierle, M., Papke, L., Revoredo, K., Matzner, M., Becker, J.: Cause vs. effect in context-sensitive prediction of business process instances. Inf. Syst. **95**, 101635 (2021)

2. Chamorro, A.E.M., Revoredo, K., Resinas, M., del-Río-Ortega, A., Santoro, F.M., Ruiz-Cortés, A.: Context-aware process performance indicator prediction. IEEE Access **8**, 222050–222063 (2020)
3. Cugola, G., Margara, A.: Processing flows of information: from data stream to complex event processing. ACM Comput. Surv. **44**(3), 1–62 (2012)
4. Ehrendorfer, M., Mangler, J., Rinderle-Ma, S.: Assessing the impact of context data on process outcomes during runtime. In: Hacid, H., Kao, O., Mecella, M., Moha, N., Paik, H. (eds.) ICSOC 2021. LNCS, vol. 13121, pp. 3–18. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-91431-8_1
5. Hütter, T., Augsten, N., Kirsch, C.M., Carey, M.J., Li, C.: JEDI: these aren't the JSON documents you're looking for? In: Management of Data, pp. 1584–1597 (2022)
6. Janiesch, C., et al.: The internet of things meets business process management: a manifesto. IEEE Syst. Man Cybern. Mag. **6**(4), 34–44 (2020)
7. Klettke, M., Störl, U., Scherzinger, S.: Schema extraction and structural outlier detection for JSON-based NoSQL data stores. In: Datenbanksysteme für Business, Technologie und Web, pp. 425–444 (2015)
8. Koschmider, A., Janssen, D., Mannhardt, F.: Framework for process discovery from sensor data. In: Enterprise Modeling and Information Systems Architectures. CEUR Workshop Proceedings, vol. 2628, pp. 32–38 (2020)
9. Levenshtein, V.I., et al.: Binary codes capable of correcting deletions, insertions, and reversals. In: Soviet Physics Doklady, pp. 707–710, no. 8. Soviet Union (1966)
10. Mangler, J., Kunkler, M.: XES logistics and transportation dataset - large (19 days) (2023). https://doi.org/10.5281/zenodo.7528638
11. Mangler, J., et al.: Datastream XES extension: embedding IoT sensor data into extensible event stream logs (2023). https://doi.org/10.3390/fi15030109
12. Mannhardt, F.: Multi-perspective process mining. In: BPM (Dissertation/Demos/Industry), pp. 41–45 (2018)
13. Park, G., Benzin, J., van der Aalst, W.M.P.: Detecting context-aware deviations in process executions. In: Di Ciccio, C., Dijkman, R., del Río Ortega, A., Rinderle-Ma, S. (eds.) BPM 2022. LNBIS, vol. 458, pp. 190–206. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-16171-1_12
14. Seiger, R., Franceschetti, M., Weber, B.: An interactive method for detection of process activity executions from IoT data. Future Internet **15**(2), 77 (2023)
15. Stertz, F., Rinderle-Ma, S., Mangler, J.: Analyzing process concept drifts based on sensor event streams during runtime. In: Fahland, D., Ghidini, C., Becker, J., Dumas, M. (eds.) BPM 2020. LNCS, vol. 12168, pp. 202–219. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-58666-9_12
16. Teinemaa, I., Dumas, M., Maggi, F.M., Di Francescomarino, C.: Predictive business process monitoring with structured and unstructured data. In: La Rosa, M., Loos, P., Pastor, O. (eds.) BPM 2016. LNCS, vol. 9850, pp. 401–417. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-45348-4_23
17. Weinzierl, S., Revoredo, K., Matzner, M.: Predictive business process monitoring with context information from documents. In: European Conference on Information Systems (2019)
18. Yeshchenko, A., Durier, F., Revoredo, K., Mendling, J., Santoro, F.: Context-aware predictive process monitoring: the impact of news sentiment. In: Panetto, H., Debruyne, C., Proper, H.A., Ardagna, C.A., Roman, D., Meersman, R. (eds.) OTM 2018. LNCS, vol. 11229, pp. 586–603. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-02610-3_33