Modeling and Using Complex IoT Time Series Data in Case-Based Reasoning: From Application Scenarios to Implementations

Lukas Malburg[1,2,*,†], Alexander Schultheis[2,†] and Ralph Bergmann[1,2]

[1]Artificial Intelligence and Intelligent Information Systems, University of Trier, 54296 Trier, Germany
[2]German Research Center for Artificial Intelligence (DFKI), Branch University of Trier, 54296 Trier, Germany

Abstract

The research area of Internet of Things (IoT) is gaining more relevance for several domains and application areas, including Case-Based Reasoning (CBR). However, IoT data is characterized by high volumes and variance of data types, making the application of CBR methods difficult. Since only few works have been published in this area so far, the integration and consideration of complex IoT data such as time series data in CBR frameworks is still in its infancy. To catch up with the current state-of-the-art, we present a comprehensive literature review on Temporal Case-Based Reasoning and time series data in CBR as part of our contribution. Furthermore, we present typical application scenarios for using IoT time series data in practice that can be addressed in further research. To build suitable CBR implementations for that purpose, we define a procedure model that can be used for time series data in CBR. In this context, we address the implementation of the application scenarios in the ProCAKE CBR framework.

Keywords

Case-Based Reasoning, Temporal Case-Based Reasoning, Internet of Things, Time Series Data, ProCAKE

1. Introduction

Temporal Case-Based Reasoning (TCBR) [1, 2] investigates how temporal relationships can be expressed in cases. A case expressing temporal relationships is a sequence of a certain attribute related to the time dimension. Recently, the Internet of Things (IoT) area is getting more and more importance in several application domains and research areas, e. g., Business Process Management (BPM) [3, 4, 5] but also in Case-Based Reasoning (CBR) [6, 7, 8, 9]. However, IoT data is characterized by very high volumes and a variety of data types [10], which makes it hard for applying classic CBR methods, e. g., for similarity assessment. Although related work has already identified some application scenarios for complex time series data like IoT data, e. g., for prediction of values in time series [11, 1, 12, 13, 14, 15, 16], in medical applications [17, 18, 19], or for classification and error-detection [20, 21, 22], research regarding this aspect is still in its

ICCBR BEAR'23: Workshop on Beyond Attribute-Value Case Representation at ICCBR2023, July 17 – 20, 2023, Aberdeen, Scotland

* Corresponding authors.
† These authors contributed equally.
malburgl@uni-trier.de (L. Malburg); alexander.schultheis@dfki.de (A. Schultheis); bergmann@uni-trier.de (R. Bergmann)
0000-0002-6866-0799 (L. Malburg); 0000-0002-2458-9943 (A. Schultheis); 0000-0002-5515-7158 (R. Bergmann)

infancy and the IoT domain is only rarely investigated in the CBR community. Even if specific concepts and ideas regarding time series data like IoT sensor data are proposed, a concrete implementation in a CBR framework is often missing [23]. The latter is also related to the fact that current CBR frameworks mostly only provide basic support for time series data, and, thus, enhanced similarity measures that go beyond simple numerical similarity measures applied to individual time points are pending. The contribution of this workshop paper is threefold: 1) we present a comprehensive literature study regarding TCBR and, in general, complex time series data used in CBR; 2) we present three application scenarios that are mainly derived from our research regarding the IoT domain [24, 25, 5, 4] and, in addition, we briefly sketch how CBR can help to utilize complex time series data; and 3) we present how the application scenarios can be concretely implemented in a state-of-the-art and open accessible CBR framework. For this purpose, we use the open-source *ProCAKE* CBR framework[1] [26] and provide a demo repository with the source code consisting of a case base with complex time series data from our smart factory [24, 25, 5, 4], a similarity model, and a vocabulary as domain representation. In addition, we present several similarity measures available in *ProCAKE* with which the similarity assessment can be performed appropriately by considering the complete structure of the time series and not only individual time points independently.

The paper is structured as follows: Sect. 2 presents the foundations for this work regarding the representation and different similarity measures for time series. In Sect. 3, we discuss related work with a special focus on TCBR and, in general, on time series. Three application scenarios that are based on our research in the IoT domain are discussed in Sect. 4. Based on that, we define a procedure model describing how a time series CBR system can be created in Sect. 5. This is demonstrated by a concrete implementation using the state-of-the-art CBR framework *ProCAKE* in Sect. 6, that shows how complex time series data can be modeled, and how the similarity can be assessed between cases. Finally, we summarize the workshop paper and provide an outlook for our planned future work in Sect. 7.

2. Foundations

In the following, we introduce the foundations for this work with IoT time series in CBR. Section 2.1 presents the approach we use to represent time series data in CBR. An overview of different similarity for time series data is given in Sect. 2.2.

2.1. Representation of Time Series Data in Case-Based Reasoning

The machine representation of time series is investigated in many research areas, especially in the field of data mining [27]. In the context of TCBR, a time series represents the simplest form of a case, among episodes [28], workflows [29], and event sequences [30]. A time series stands for a measured real value over a time course [31, 32], where especially concrete time points are referred [33]. In the IoT domain, such time points can be contained in sensor data, for example [24]. To represent the time series, we use a symbolic representation, which can vary depending on the use case, and which represents the real values. These values are summarized

[1] https://procake.uni-trier.de

and mirrored as a feature vector [34]. This can be embedded in other objects at higher levels as desired. The individual values within the sequence are also objects that contain, on the one hand, the timestamp and, on the other hand, the symbolically represented value at this time.

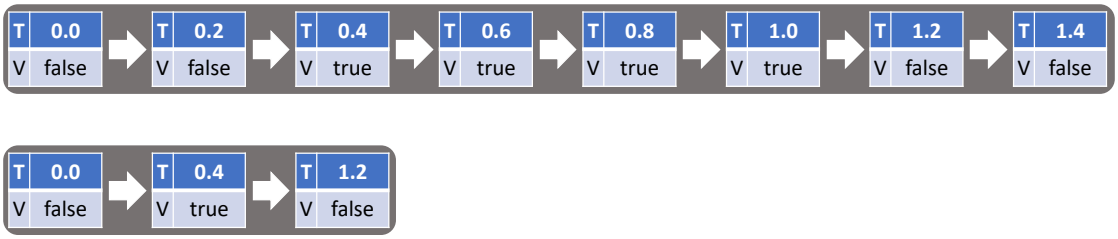

Figure 1: Two Exemplary Time Series That Represents Boolean Values Over a Time-Period. The Second Time Series Is Compressed Compared to the First, but Contains the Same Values.

Figure 1 shows an example of a time series represented in this way. In this specific example, boolean values are captured that mirror the interruption of a light barrier. One time series contains the values at 0.2 millisecond intervals, the other contains only the times at which the values change. This time series contains for each time a pair of the time stamp and the corresponding value, which are stored in an ordered sequence for processing in CBR.

2.2. Similarity Measures for Time Series Data

In the following, we introduce several similarity measures that can be used to compute similarities between sequences, and, thus, between time series. We divide the similarity measures into three categories:

Cat. 1 Similarity measures that can only be applied to time series of the same length. These compare only the values at the corresponding times.

Cat. 2 Similarity measures that can be applied to time series of different lengths and consider not only the values at the time points, but the time points themselves.

Cat. 3 Similarity measures like in Cat. 2, that can detect stretching and compression in addition.

The similarity calculation for these measures is based on the local-global principle [35]. In this context, similarities are calculated at the level of individual attributes and then aggregated globally for the complete case. In the following, the similarity measures that can be applied to the above categories are presented.

Similarity Measures for Cat. 1: When comparing two sequences, which are of the same length, a direct mapping of the individual elements to each other is performed. Only the values at the respective time points are compared. In the literature, approaches that use the Euclidean distance to calculate these similarities are often used for this purpose [36]. Other approaches from literature are based on the Hamming [37, 38] or the Levenshtein distance [39, 40]. In general, any similarity measure suitable for the domain can be used to calculate the similarities between the elements of the sequence and then aggregated. However, comparing time series of the exactly same length in CBR is rather unsuitable because the query typically includes only a

fraction of an entire time series. In this context, no similarity can be calculated. For the used example presented in Fig. 1, no similarity calculation can be performed due to the different lengths, so the similarity value would be 0.0.

Similarity Measures for Cat. 2: To deal with sequences of different lengths, other similarity measures exist. The simplest algorithm consists of sequential matching, as in Cat. 1, where elements that cannot be mapped to a corresponding element of the other time series are assessed with a local similarity of 0.0. One more complex algorithm, based on the idea of the Levenshtein distance, is the *Smith-Waterman-Algorithm* (SWA) [41]. It determines the matching of sequences based on the required insertion or deletion operations [41]. This is done based on a scoring matrix in which operations are scored negatively so that the best possible similarity value can be determined from the matrix. Other measures are based on the well-known *Longest Common Subsequence* (LCS) problem [42, 43], which can also be applied to time series. The LCS is searched [42] for, and the similarity is determined based on this. Various algorithms for this exist [43]. Again, in the algorithms described, any local similarity measure like the ones named for Cat. 1 can be used at the local level to determine mappings or to determine the longest subsequence. However, none of the described algorithms can deal with stretched or compressed time series. In the example shown in Fig. 1, the second time series is the compressed version of the first one or respectively the first one is the stretched version of the second one. If such kinds of time-series appear, the similarity is low, even though the time series have the same trend but are represented differently, e. g., with a higher or lower sampling rate. In the literature [44], SWA has been identified as one of the most promising measures, so we consider it as an example for Cat. 2. Here, we refer to the version of Schake et al. [44] who extended this by using local similarity functions for operations between elements. For the example from Fig. 1, the penalties for inserting or deleting elements lead to a low similarity value. With equal penalties for both operations, it would not matter which of the two time series is used as a request and which as a case. The fact that both time series contain the same values is not recognized. When suspending the penalties for the insert and delete operations, the similarity could be maximized, but in return, the measure would also recognize many other dissimilar time series as similar.

Similarity Measures for Cat. 3: Similarity measures also exist to consider shifts, compression, and stretching in time series. Thus, these measures can also compensate missing values, as long as no content information is lost. The *Dynamic Time Warping* (DTW) approach allows elements of the sequences to be mapped on warped elements, thus, preventing compression and stretching [45, 46]. Analogous to SWA, scoring matrices are created that determine the steps from one sequence to another. The maximum value is the best possible similarity. The *Minimum Jump Costs* (MJC) maps two sequences onto each other, and forward jumps can be made from individual points of one to the other [47]. These must be kept to a minimum to map the sequences onto each other in the best possible way. A similarity value can be derived based on these distances. Another similarity measure with a different approach is the *Weighted Vector Similarity* (WVS), which transforms the sequences into a vector representation [48]. This measure has been originally developed for event sequences, so the similarity is calculated based on the end of these events and their importance. On this basis, the event sequence is transformed into a vector representation, so that similarity measures such as cosine similarity can be used between the vectors. In the literature [44], DTW has also been identified as one of the most promising measures, so we consider it as an example for this category. Again, we

refer to the extended version of Schake et al. [44], who extended this measure by using local similarity functions for operations between elements. For the example presented in Fig. 1, the measure recognizes that the second time series is a compressed version of the first one, or that the first time series is a stretched time series of the second one. Unlike SWA, necessary insertions or deletions are not penalized if values are repeated. Based on a suitable comparison of the time points that recognizes that nothing has been moved in time, the measure recognizes the two time series as equal and returns 1.0 as similarity value.

3. Related Work

In the following, we present as part of our contribution a comprehensive literature review w. r. t. TCBR and, in general, time series in CBR. All approaches have in common that they use complex (IoT) data as time series for certain application scenarios. Although some of these works go beyond simply describing a possible implementation, it usually remains unclear how the concept can be implemented and how other researchers can reproduce and reuse concrete implementations.

First, an overview of general work on time series in CBR is given (see Sect. 3.1). During the literature search on TCBR and the use of time series in CBR, we found several publications that can be divided into three categories: 1) Prediction based on Time Series (see Sect. 3.2), 2) Medical Use of Time Series (see Sect. 3.3), and 3) Classification and Error Detection (see Sect. 3.4). In addition, various publications are found that deal with only one topic (see Sect. 3.5).

3.1. General Contributions to Time Series in Case-Based Reasoning

A first approach to time-extended cases is presented by Jaczynski [49]. During similarity assessment, only single time points are used and not the full context of the time series. As example domains, Jaczynski lists automatic control and process supervision. Ma and Knight [50] deal with historical CBR, where they examine case histories. A case history is a collection of time-independent elemental cases. The computation of similarity between them is done twice, once time-dependent based on stored time reference points and once time-independent based on standard cases. Sànchez-Marrè et al. [28] define the requirement for TCBR to be dynamic and continuous in solving cases, and to consider temporal dependencies. They store cases as episodes, which in turn consist of several actual cases that are already included in the case base. Additionally, meta information can be stored. Monitoring and on-line process control are mentioned as example applications. Montani et al. [51] implement a framework that supports retrieval with time series data. Temporal abstraction is used to summarize time series features. In later work [52], they extend the framework to support subsequence matching. All in all, these presented approaches are frequently referenced in literature in the context of TCBR.

Fuad and Marteau [53] address the problem of runtime complexity in retrieving cases with time series data. For this purpose, they propose a reduction of dimensionality by applying multi-resolution techniques. These are methods that speed up similarity search by improving distance computations. Lupiani et al. [54] define cases as temporal event sequences and present five methods for improving retrieval efficiency, which are applicable as long as the case base contains only temporal cases.

3.2. Prediction based on Time Series in Case-Based Reasoning

Nakhaeizadeh [11] uses time series data in CBR to predict following values of the time series. In this context, no case representation for time series is proposed, as the CBR methodology should be used to predict next values based on previous cases. The work of Jære et al. [1] goes into a similar direction by applying CBR to predict upcoming problems based on previous time series data. For this purpose, they utilize an interval-based approach. Compta and López [12] extract sequences from log data and transfer them to cases in a CBR system. Based on this, they predict other log values for a system. Gay et al. [55] generate sequences from events and use CBR to detect patterns in them and, based on them, predict further events. Platon et al. [13] use a CBR system to predict the hourly energy consumption of buildings given as time series. In this context, the CBR system continuously learns the new cases and, thus, improves its accuracy. A similar approach is also used by Shabani et al. [14]. Ihle [15] also uses CBR to predict energy consumption for one day at a time on a time series basis. Dolphin et al. [16] use financial time series to make predictions on the stock market.

3.3. Medical Use of Time Series in Case-Based Reasoning

In the medical domain, Funk and Xiong [17] use sequences to represent time series cases in CBR, on which basis a classification is performed. Similarly, Nilsson et al. [18] identify stress-related disorders based on time series. Szczepanski et al. [19] evaluate data from patients with back pain, which are represented in the form of time series, to make recommendations for immediate or long-term therapy based on the most similar cases. Several challenges in the use of time series in CBR are outlined, which will be addressed in the future.

3.4. Classification and Error-Detection based on Time Series in Case-Based Reasoning

Other works exist that address categorization and error detection based on time series in CBR. For example, Fritsche et al. [20] use it to identify critical situations. In this context, current time series are compared with cases based on DTW and possible failure cases are determined based on that. Borck et al. [21] use time series in CBR to monitor astronauts. In this work, CBR is used to determine which task is currently being performed by them and whether errors occur. The work of Ariza et al. [22] addresses the detection of skill levels of players based on CBR. Cases in this context are time series that measure multiple parameters of the player during the game and classify the player into a level based on that.

3.5. Other Applications of Time Series in Case-Based Reasoning

Time series in CBR are used in further applications in individual papers. Elsayed et al. [56] represent images in form of time series representing curves and categorize them using CBR. Zarka et al. [57] work with episodes as cases in which user interactions with a computer system are stored. Minor and Marx [58] use time series in CBR for energy management to prevent energy waste. The representation of the time series data is either based on set-point values, measured values, or disturbance values. The phases of retrieval and reuse are discussed.

Valdez-Ávila et al. [59] use CBR as a method to explain results of a predictive model based on time series.

4. Application Scenarios for Complex IoT Time Series Data in Case-Based Reasoning

In the following, we present three application scenarios (**AS1**-**AS3**) from our previous work [60] in the context of predictive maintenance (see **AS1**) and planned future work regarding the use of CBR for event and activity detection (see **AS2**) and for detection of data quality issues (see **AS3**). We briefly sketch how CBR can be used in these application scenarios.

AS1 *Predictive Maintenance:* The aim of using predictive maintenance [60], is to determine errors or faults during runtime of components or machines in industrial settings before the fault itself occurs. Typically, the remaining useful lifetime is determined that specifies how long a machine can be used before the fault appears. In current research, the use of data-driven predictive maintenance by applying deep neural networks is common (see [60] for an overview) to fulfill this task. Klein et al. [60] present an approach for predictive maintenance with Siamese neural networks in combination with expert knowledge from a CBR system to better classify faults. In this application scenario, we want to focus on the use of pure CBR methods so that it is not needed to train a deep neural model for data-driven predictive maintenance. For this purpose, cases consisting of the observed time series and the remaining useful lifetime are stored in a case base. During retrieval, a time series similarity measure is applied for determining the most useful case from which the remaining lifetime is used.

AS2 *Event and Activity Detection:* In Industrial Internet of Things (IIoT) settings, a high volume of complex time series data of sensors is available. By using BPM methods in IoT [3], one research area is the identification of activities from executed process instances by using the IoT data. Malburg et al. [25] present a multi-modal model for event and activity detection consisting of Complex Event Processing (CEP) methods that are applied for deriving higher-level events from sensor data and object detection methods for processing video data. Besides this, it is also possible to apply CBR for processing sensor data to detect events and activities during process execution. For this purpose, it is necessary to store cases that consist of typical sensor patterns needed to identify the event or activity. This process is typically a knowledge-intensive and demanding task that is done manually by a domain expert in an interactive fashion [61]. For this reason, it is important to store the created sensor patterns of domain experts as cases in a case base and to utilize them in similar situations to remedy the high efforts.

AS3 *Data Quality Issue Detection:* Data from IoT sensors can be faulty, e. g., the sensor is not correctly calibrated, or the sensor does not provide any data since there are connection issues. As the latter type of error can be detected easily in the corresponding data stream, it is hard to identify quality issues resulting from not correctly calibrated sensors or other issues coming from incorrect data records at all. Fixing these data quality issues is a demanding, mainly manually performed task [62] but crucial, as incorrect and faulty data can lead to improper decisions made based on them. In addition, if the IoT data is used in context of BPM, e. g., for

generating event logs for process mining [24, 63], incorrect logs are produced and cannot be used for analyses. The aim of detecting data quality issues as soon as possible in near real-time is to immediately address the problem, resulting in correct, error-free data, and, thus, correct event logs. Similar to **AS2**, CBR can be used to detect issues in time series data in near real-time. For this purpose, a case base can be created in which similar problem situations with already detected faults are stored. During runtime, queries are generated, and similar cases are retrieved from the case base. If a similar case is found, the solution of the case determines whether the current time series data from the IoT sensors is faulty or not and, if it is the case, how this fault effects resulting event logs or other higher-level systems.

5. Procedure Model for Creating a CBR System

For the application of complex time series data in CBR systems, we define a five-step procedure model that describes, in general, how unprocessed data can be transformed into a suitable, working CBR implementation. Therefore, the four CBR knowledge containers, vocabulary, case base, similarity measures, and adaptation knowledge according to Richter [64] are used as foundation. The proposed procedure model is shown in Fig. 2, which is inspired by the CRISP-DM model [65]. In the following, the five steps of the model are described in detail.

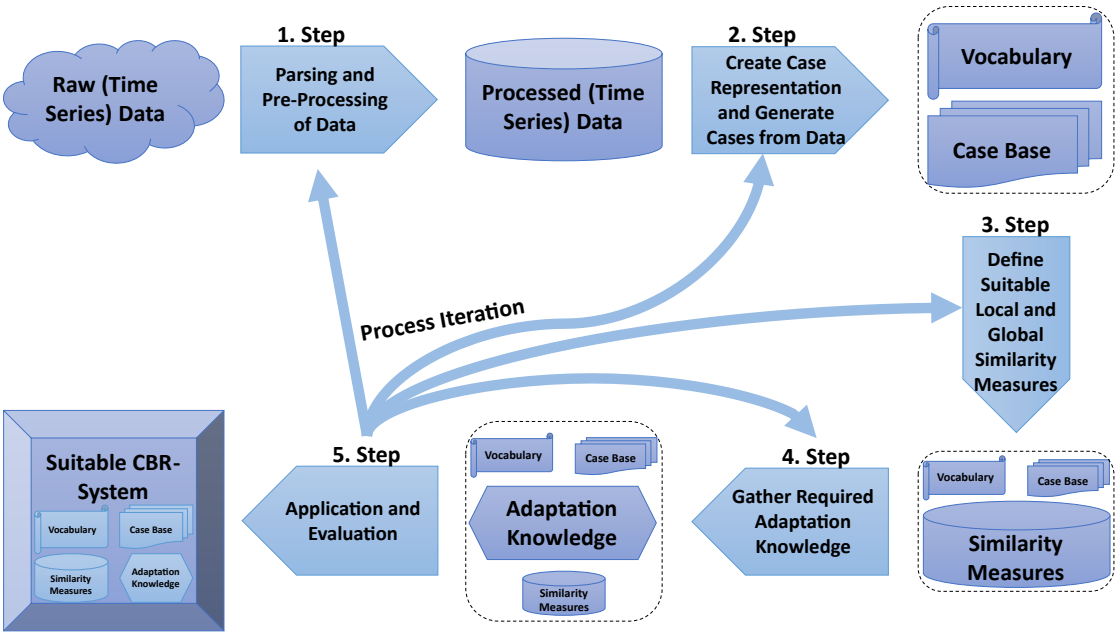

Figure 2: A Procedure Model That Shows How to Build a Working and Suitable CBR Application From (Time Series) Raw Data in Five Steps.

1. Parsing and Pre-Processing of Data: In this step, the data to be considered is selected and pre-processed. For example, redundant log data can be removed. Furthermore, check for missing values takes place here, as well as any appropriate actions that may be necessary to resolve this issue. The data is read in after this step, but is not yet in the correct format.

2. Create Case Representation and Generate Cases from Data: To be able to process the data in the desired use case with CBR methods, the knowledge container of the vocabulary (cf. [64]) is built up first. There is defined which structure the cases (and possible queries) should have. Subsequently, the read-in and pre-processed data is transferred into this defined format by a converter. By this, the knowledge container of the case base (cf. [64]) is created.

3. Define Suitable Local and Global Similarity Measures: Suitable local and global similarity measures are defined, which form the knowledge container of the similarity knowledge (cf. [64]). These are explicitly tailored to the vocabulary at hand and the domain from which the data was read.

4. Gather Required Adaptation Knowledge: Last, the fourth CBR knowledge container of adaptation knowledge (cf. [64]) is considered. For this, knowledge is collected with which a domain-dependent adaptation can be carried out. If this is already covered by a sufficiently large case base and good similarity measures, or if the domain only allows null adaptation, this step can be skipped.

5. Application and Evaluation: By integrating the four defined knowledge containers in a CBR framework, a CBR system can be created. This must be evaluated for its suitability to the domain and the desired use case. If the CBR system does not deliver the desired results, it is possible to go back to one of the previous steps. If e. g., attributes are missing, it is possible to go back to step 1, in case of errors in the three knowledge containers to step 2, step 3, or step 4.

6. Implementation of Complex IoT Time Series Data in ProCAKE

Based on the model presented in Sect. 5, a CBR system can be designed that is capable of handling time series data for an application scenario. In the following, we present an implementation created with our domain-independent CBR framework ProCAKE [26] that focuses on structural and process-oriented CBR. ProCAKE provides a generic data type model for custom case representations, various syntactic and semantic similarity measures, and several retrieval algorithms. Furthermore, several similarity measures for dealing with sequence data such as complex time series are provided.

6.1. Parsing and Pre-Processing of Data

The data used for this implementation comes from the Fischertechnik Smart Factory[2] [66, 5], which is developed and maintained by the University of Trier and the German Research Center for Artificial Intelligence at the Trier Branch. In this smart factory, various sensors are installed to measure and store the behavior of all its components. The data is stored in the DataStream format [67], which is an extension of the XES (eXtensible Event Stream) [68] standard. The corresponding data set in this format [69] is freely available online [3]. As there is no missing data in this data set, no corresponding actions need to take place. To read in the data and process it, it was necessary to develop a converter (see 1. Step in Fig. 2).

[2] https://iot.uni-trier.de
[3] https://zenodo.org/record/7795547

The timestamps of the sensor values are removed during the preprocessing of the data. The start time of a time series is always set to 0 and all subsequent elements of the time series contain as time value the distance to the start point in milliseconds. Since the measured sensor values are very close in time, the component of when exactly the execution has been carried out can be omitted. As further pre-processing by the converter, a compression of the individual sensor data is performed. Instead, sensor values that occurred unchanged multiple times in a row have been removed so that only the first occurrence of the value, including the timestamp, is further considered. In the example from Sect. 2.1 all intermediate values with the boolean expressions "true" and "false" are deleted, so that only the times of the first breaking or releasing of the light barrier are included. Compared to an iteration of all model steps without this truncation, the runtime is reduced considerably without changing the retrieval results.

6.2. Create Case Representation and Generate Cases from Data

To be able to store the imported data in ProCAKE as cases, it is necessary to design a data model (see 2. Step in Fig. 2). In ProCAKE, data classes can be created both during runtime in Java and in advance in XML. In the following, we describe to the necessary extent the created data classes and their storage in XML format. To transfer the data into the ProCAKE-specific format, corresponding domain-specific methods are available for the converter used in the previous step.

The individual values of the time series are stored using an `Aggregate` object, that can combine different values identified by a specific attribute name. For each attribute, allowed data classes must be specified in the data model. For every time series in this domain, the first value is the time value in milliseconds normalized in the previous step, which is stored from the beginning of the sensor measurement. For the example presented in Sect. 2.1, which contains information about the interruption of a light barrier, a boolean object is used. The definition of this data class is represented as follows:

Listing 1: Definition of Data Class in ProCAKE.

```
<AggregateClass name="TimeBooleanPair">
   <Attribute name="Timestamp" class="Double"/>
   <Attribute name="Boolean" class="Boolean"/>
</AggregateClass>
```

Analogously, other values are also stored. For example, many sensors record position values, whereby only the changed values are considered here as well. For this, numerical values are used for the second attribute, which are represented similarly as already presented in Listing 1.

These individual values can be combined in ProCAKE by a list, so that a sequence of these values is created. For this purpose, corresponding objects are created, each of which contains one of the local classes for defining the pairs. For the recorded values about the interruption of light barriers, the class for summarizing as a time series is represented in Listing 2.

Listing 2: Definition of Time Series in ProCAKE.

```
<ListClass name="BooleanList">
   <ElementClass name="TimeBooleanPair"/>
</ListClass>
```

These individual time series classes are combined into a higher-level aggregate class. This contains several lists representing time series for position values and time series for interrupted light barriers, as well as additional information from service execution. For reasons of scope, this global class is not presented here.

Based on this presented data model, a conversion of the data read in and pre-processed in the first step into ProCAKE-specific cases is performed by the converter. The generated case base is also serialized into an XML file so that it is directly available for future applications. If required, this can also be done directly during runtime. For our data set, 4,847 cases have been generated, each of which are objects of the top aggregate class. These contain nested objects of the respective subclasses. Based on this, the vocabulary and case base are provided in XML format as the result of the second step (see Fig. 2).

6.3. Define Suitable Local and Global Similarity Measures

ProCAKE contains various similarity measures that must be initialized via XML or at runtime for a specific data class and can then be used. The similarity measures must be defined starting from the local level up to the top global level. In our scenario, one measure is needed at the level of the individual element, one for the time series and one for the object above. For reasons of relevance, only the similarity measures for the time series themselves are discussed here.

At the local level, a distance-based measure is used to compare the timestamps. In this case, it is a measure that uses a linear function to assign a lower similarity value as the distance increases, so that, the similarity drops constantly up to a total distance of one minute, after which the value remains at 0. The measure can be defined as depicted in Listing 3.

Listing 3: Similarity Measure for Time Dimension.

```
<NumericLinear name="SMTime" class="Double" min="0.0" max
   ="60000.0" default="true"/>
```

For the comparison of boolean values, a similarity measure is used that is based purely on the equality of the values used. This is defined as illustrated in Listing 4.

Listing 4: Similarity Measure for Time Dimension.

```
<ObjectEqual name="SMBoolean" class="Boolean"
default="true"/>
```

At the level above, these measures are used by a similarity measure suitable for lists. ProCAKE has a total of four classic measures for comparing collections and two measures explicitly intended for comparing sequences. In Sect. 2.2, three categories of similarity measures for time series data are introduced. By the similarity measure `ListMapping`, the measures from

Cat. 1 can be mapped, and corresponding local measures must be defined. For Cat. 2, the SWA measure is currently explicitly implemented, for Cat. 3 the DTW measure. SWA and DTW are represented for lists, as well as for sequences represented as workflows. In past research [44], these two measures have already been applied to sequences using ProCAKE. In the example domain described, the DTW measure is applied because it can handle compression and stretching and, thus, is most likely to find the best mapping. This is instantiated as follows.

Listing 5: Similarity Measure for Dynamic Time Wraping.

```
<ListDTW name="SMBooleanList" class="BooleanList"
localSimName="SMTimeBooleanPair"/>
```

The global measure was created as `AggregateAverage`. These similarity measures are stored in an XML file.

For the described application scenarios **AS1**, **AS2** and **AS3** no adaptation knowledge is needed because only a null-adaptation may be required. Thus, for our scenario, the step of acquiring the adaptation knowledge can be omitted.

6.4. Application and Evaluation

Based on the previous four steps, the four knowledge containers are filled so that an executable CBR system can be created using the framework passed through by ProCAKE as the CBR framework. The code for this sample application is published as part of ProCAKE's demo project and is available online[4]. In this system, exemplary queries are made in the context of **AS2**, for which sensor data with the activity description removed is used. A retrieval is performed based on these queries, which returns plausible cases as most similar ones. For some queries, cases are returned that contained other, very similar activities. The time required for retrieval is a few hundreds of milliseconds, typically between 100 and 200, on a standard computer[5] at the case base size mentioned, so efficient retrieval is possible. In one iteration, these times could be optimized, which were about 30 to 40 seconds before the described preprocessing of the data (see Sect. 6.1).

Thus, the implementation of the presented model is shown. Since the modeling is very complex and differs depending on the use case, we only present the implementation for **AS2** here. The functionality of the CBR system is similar for each use case, so the implementation would be very analogous to **AS1** and **AS3**.

7. Summary and Outlook

In this paper, we present a comprehensive literature review about related approaches that use time series data in CBR. One major drawback about the related approaches is that often concepts are described but not implemented. For this reason, we discuss typical application scenarios for using complex time series data in the IoT domain and based on that, we present a

[4] https://gitlab.rlp.net/procake/procake-demos, executable class: `de.uni_trier.wi2.procake_demos.timeSeries.TimeSeriesDemo.java`

[5] We refer to a computer with 32 GB of RAM, an i7 processor with six cores and twelve logical processors.

prototypical implementation for using time series in our developed ProCAKE CBR framework. Currently, we only implemented one of the presented application scenarios and this scenario has currently not been evaluated intensively. In future work, we want to address these aspects. In addition, we plan to implement the other two scenarios for our Fischertechnik smart factory. In this context, we want to investigate how CBR can be applied for processing time series data directly at the edge or whether it is necessary to handle it in a monolithic cloud CBR system.

Acknowledgments. This work is funded by the Federal Ministry for Economic Affairs and Climate Action under grant No. 01MD22002C *EASY*.

References

[1] M. D. Jære, A. Aamodt, P. Skalle, Representing Temporal Knowledge for Case-Based Prediction, in: 6th ECCBR Proc., volume 2416 of *LNCS*, Springer, 2002, pp. 174–188.

[2] B. López, Case-Based Reasoning: A Concise Introduction, Synth. Lect. Artif. Intell. Mach. Learn., Morgan & Claypool Publishers, 2013.

[3] C. Janiesch, A. Koschmider, M. Mecella, B. Weber, A. Burattin, C. D. Ciccio, A. Gal, U. Kannengiesser, F. Mannhardt, J. Mendling, A. Oberweis, M. Reichert, S. Rinderle-Ma, W. Song, J. Su, V. Torres, M. Weidlich, M. Weske, L. Zhang, The Internet-of-Things Meets Business Process Management. A Manifesto., IEEE Syst. Man. Cybern. Mag. 6 (2020) 34–44.

[4] L. Malburg, F. Brand, R. Bergmann, Adaptive Management of Cyber-Physical Workflows by Means of Case-Based Reasoning and Automated Planning, in: 26th EDOC Workshops, volume 466 of *LNBIP*, Springer, 2023, pp. 79–95.

[5] R. Seiger, L. Malburg, B. Weber, R. Bergmann, Integrating Process Management and Event Processing in Smart Factories: A Systems Architecture and Use Cases, J. Manuf. Syst. 63 (2022) 575–592.

[6] D. W. Yun, W. C. Lee, Intelligent Dynamic Real-Time Spectrum Resource Management for Industrial IoT in Edge Computing, Sensors 21 (2021) 7902.

[7] M. F. Valdez-Ávila, G. A. Pérez-Pérez, H. Sarabia-Osorio, C. Bermejo-Sabbagh, M. G. Orozco-del Castillo, A Case-based Explanation Method for Weather Forecasting, in: 30th ICCBR Workshop Proc., volume 3389, CEUR-WS.org, 2022, pp. 153–164.

[8] C. Pirie, Explaining and Upsampling Anomalies in Time-Series Sensor Data, ICCBR Doctoral Consortium 2022 1613 (2022) 0073.

[9] J. A. Recio-Garcia, B. Diaz-Agudo, A. Acuaviva-Huertos, Becalm: Intelligent Monitoring of Respiratory Patients, IEEE J. Biomed. Health Inform. (2023).

[10] A. Gaddam, T. Wilkin, M. Angelova, J. Gaddam, Detecting Sensor Faults, Anomalies and Outliers in the Internet of Things: A Survey on the Challenges and Solutions, Electronics 9 (2020).

[11] G. Nakhaeizadeh, Learning Prediction of Time Series – A Theoretical and Empirical Comparison of CBR with some other Approaches, in: 1st EWCBR Proc., volume 837 of *LNCS*, Springer, 1993, pp. 65–76.

[12] M. Compta, B. López, Integration of Sequence Learning and CBR for Complex Equipment Failure Prediction, in: 19th ICCBR Proc., volume 6880 of *LNCS*, Springer, 2011, pp. 408–422.

[13] R. Platon, J. Martel, K. Zoghlami, CBR Model for Predicting a Building's Electricity Use:

On-Line Implementation in the Absence of Historical Data, in: 23rd ICCBR Proc., volume 9343 of *LNCS*, Springer, 2015, pp. 306–319.

[14] A. Shabani, A. Paul, R. Platon, E. Hüllermeier, Predicting the Electricity Consumption of Buildings: An Improved CBR Approach, in: 24th ICCBR 2016 Proc., volume 9969 of *LNCS*, Springer, 2016, pp. 356–369.

[15] N. Ihle, Case Representation and Adaptation for Short-Term Load Forecasting at a Container Terminal, in: 31st ICCBR Workshop Proc., volume 1815, CEUR-WS.org, 2016, pp. 142–151.

[16] R. Dolphin, B. Smyth, Y. Xu, R. Dong, Measuring Financial Time Series Similarity with a View to Identifying Profitable Stock Market Opportunities, in: 29th ICCBR Proc., volume 12877 of *LNCS*, Springer, 2021, pp. 64–78.

[17] P. Funk, N. Xiong, Case-Based Reasoning and Knowledge Discovery in Medical Applications with Time Series, Comput. Intell. 22 (2006) 238–253.

[18] M. Nilsson, P. J. Funk, E. M. G. Olsson, B. von Schéele, N. Xiong, Clinical Decision-Support for Diagnosing Stress-Related Disorders by Applying Psychophysiological Medical Knowledge to an Instance-Based Learning System, Artif. Intell. Medicine 36 (2006) 159–176.

[19] T. Szczepanski, K. Bach, A. Aamodt, Challenges for the Similarity-Based Comparison of Human Physical Activities Using Time Series Data, in: 31st ICCBR Workshop Proc., volume 1815, CEUR-WS.org, 2016, pp. 173–177.

[20] L. Fritsche, A. Schlaefer, K. Budde, K. Schröter, H. Neumayer, Research Paper: Recognition of Critical Situations from Time Series of Laboratory Results by Case-Based Reasoning, J. Am. Medical Informatics Assoc. 9 (2002) 520–528.

[21] H. Borck, S. Johnston, M. Southern, M. S. Boddy, Exploiting Time Series Data for Task Prediction and Diagnosis in an Intelligent Guidance System, in: 31st ICCBR Workshop Proc., volume 1815, CEUR-WS.org, 2016, pp. 132–141.

[22] D. S. L. Ariza, A. A. Sánchez-Ruiz, P. A. González-Calero, Time Series and Case-Based Reasoning for an Intelligent Tetris Game, in: 25th ICCBR Proc., volume 10339 of *LNCS*, Springer, 2017, pp. 185–199.

[23] A. Schultheis, C. Zeyen, R. Bergmann, An Overview and Comparison of Case-Based Reasoning Frameworks, in: 31st ICCBR Proc., LNCS, Springer, 2023. Accepted for Publication.

[24] L. Malburg, J. Grüger, R. Bergmann, An IoT-Enriched Event Log for Process Mining in Smart Factories, CoRR abs/2209.02702 (2022).

[25] L. Malburg, M. Rieder, R. Seiger, P. Klein, R. Bergmann, Object Detection for Smart Factory Processes by Machine Learning, Procedia Comput. Sci. 184 (2021) 581–588.

[26] R. Bergmann, L. Grumbach, L. Malburg, C. Zeyen, ProCAKE: A Process-Oriented Case-Based Reasoning Framework, in: 27th ICCBR Workshop Proc., volume 2567, CEUR-WS.org, 2019, pp. 156–161.

[27] J. Lin, E. J. Keogh, S. Lonardi, B. Y. Chiu, A Symbolic Representation of Time Series, With Implications for Streaming Algorithms, in: 8th ACM SIGMOD Proc., ACM, 2003, pp. 2–11.

[28] M. Sànchez-Marrè, U. Cortés, M. Martínez, J. Comas, I. Rodríguez-Roda, An Approach for Temporal Case-Based Reasoning: Episode-Based Reasoning, in: 6th ICCBR Proc., volume 3620 of *LNCS*, Springer, 2005, pp. 465–476.

[29] M. Minor, S. Montani, J. A. Recio-García, Process-Oriented Case-Based Reasoning, Inf. Syst. 40 (2014) 103–105.

[30] Z. Huang, J. M. Juarez, H. Duan, H. Li, Length of Stay Prediction for Clinical Treatment

Process Using Temporal Similarity, Expert Syst. Appl. 40 (2013) 6330–6339.
[31] D. Gunopulos, G. Das, Time Series Similarity Measures and Time Series Indexing, in: SIGMOD'01 Proc., ACM, 2001, p. 624.
[32] C. Cassisi, P. Montalto, M. Aliotta, A. Cannata, A. Pulvirenti, Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining, DMKD (2012) 71–96.
[33] J. F. Allen, Maintaining Knowledge about Temporal Intervals, Commun. ACM 26 (1983) 832–843.
[34] S. H. El-Sappagh, M. Elmogy, Case Based Reasoning: Case Representation Methodologies, Int. J. Adv. Comput. Sci. Appl. 6 (2015) 192–208.
[35] R. Bergmann, Experience Management: Foundations, Development Methodology, and Internet-Based Applications, volume 2432 of *LNCS*, Springer, 2003.
[36] G. E. A. P. A. Batista, E. J. Keogh, O. M. Tataw, V. M. A. de Souza, CID: An Efficient Complexity-Invariant Distance for Time Series, DMKD 28 (2014) 634–669.
[37] R. W. Hamming, Error Detecting and Error Correcting Codes, BSTJ 29 (1950) 147–160.
[38] A. Apostolico, C. Guerra, G. M. Landau, C. Pizzi, Sequence Similarity Measures Based on Bounded Hamming Distance, Theor. Comput. Sci. 638 (2016) 76–90.
[39] V. I. Levenshtein, Binary Codes Capable of Correcting Deletions, Insertions, and Reversals, in: Soviet Physics Doklady, volume 10, Soviet Union, 1966, pp. 707–710.
[40] P. Marteau, Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching, IEEE Trans. Pattern Anal. Mach. Intell. 31 (2009) 306–318.
[41] T. F. Smith, M. S. Waterman, et al., Identification of Common Molecular Subsequences, JMB 147 (1981) 195–197.
[42] D. S. Hirschberg, Algorithms for the Longest Common Subsequence Problem, J. ACM 24 (1977) 664–675.
[43] A. Abboud, A. Backurs, V. V. Williams, Tight Hardness Results for LCS and Other Sequence Similarity Measures, in: V. Guruswami (Ed.), 56th IEEE Proc., IEEE CS, 2015, pp. 59–78.
[44] E. Schake, L. Grumbach, R. Bergmann, A Time-Series Similarity Measure for Case-Based Deviation Management to Support Flexible Workflow Execution, in: 28th ICCBR Proc., volume 12311 of *LNCS*, Springer, 2020, pp. 33–48.
[45] H. Sakoe, S. Chiba, Dynamic Programming Algorithm Optimization for Spoken Word Recognition, IEEE Trans. Aco. Sp. and Sig. Proc. 26 (1978) 43–49.
[46] D. J. Berndt, J. Clifford, Using Dynamic Time Warping to Find Patterns in Time Series, in: KDD workshop, volume 10, Seattle, WA, USA, 1994, pp. 359–370.
[47] J. Serrà, J. L. Arcos, A Competitive Measure to Assess the Similarity Between Two Time Series, in: 20th ICCBR Proc., volume 7466 of *LNCS*, Springer, 2012, pp. 414–427.
[48] O. E. Gundersen, Toward Measuring the Similarity of Complex Event Sequences in Real-Time, in: 20th ICCBR Proc., volume 7466 of *LNCS*, Springer, 2012, pp. 107–121.
[49] M. Jaczynski, A Framework for the Management of Past Experiences with Time-Extended Situations, in: 6th CIKM Proc., ACM, 1997, pp. 32–39.
[50] J. Ma, B. Knight, A Framework for Historical Case-Based Reasoning, in: 5th ICCBR Proc., volume 2689 of *LNCS*, Springer, 2003, pp. 246–260.
[51] S. Montani, G. Leonardi, A. Bottrighi, L. Portinale, P. Terenziani, Supporting Flexible, Efficient, and User-Interpretable Retrieval of Similar Time Series, IEEE Trans. Knowl. Data Eng. 25 (2013) 677–689.

[52] A. Bottrighi, G. Leonardi, S. Montani, L. Portinale, P. Terenziani, A Time Series Retrieval Tool for Sub-Series Matching, Appl. Intell. 43 (2015) 132–149.
[53] M. M. M. Fuad, P. Marteau, Fast Retrieval of Time Series Using a Multi-resolution Filter with Multiple Reduced Spaces, in: 6th ADMA Proc., volume 6440 of *LNCS*, Springer, 2010, pp. 137–148.
[54] E. Lupiani, J. M. Juarez, J. T. Palma, A Proposal of Temporal Case-Base Maintenance Algorithms, in: 22nd ICCBR Proc., volume 8765 of *LNCS*, Springer, 2014, pp. 260–273.
[55] P. Gay, B. López, J. Meléndez, Sequential Learning for Case-Based Pattern Recognition in Complex Event Domains, in: 16th UKCBR Workshop Proc., 2011, pp. 46–55.
[56] A. Elsayed, M. H. A. Hijazi, F. Coenen, M. García-Fiñana, V. Sluming, Y. Zheng, Time Series Case Based Reasoning for Image Categorisation, in: 19th ICCBR Proc., volume 6880 of *LNCS*, Springer, 2011, pp. 423–436.
[57] R. Zarka, A. Cordier, E. Egyed-Zsigmond, L. Lamontagne, A. Mille, Similarity Measures to Compare Episodes in Modeled Traces, in: 21st ICCBR Proc., volume 7969 of *LNCS*, Springer, 2013, pp. 358–372.
[58] M. Minor, L. Marx, Case-Based Reasoning for Inert Systems in Building Energy Management, in: 25th ICCBR Proc., volume 10339 of *LNCS*, Springer, 2017, pp. 200–211.
[59] M. F. Valdez-Ávila, G. A. Pérez-Pérez, H. Sarabia-Osorio, C. Bermejo-Sabbagh, M. G. Orozco-del-Castillo, CBR-foX: A Generic Post-Hoc Case-Based Reasoning Method for The Explanation of Time-Series Forecasting, in: 30th ICCBR Workshop Proc., volume 3389, CEUR-WS.org, 2022, pp. 246–250.
[60] P. Klein, N. Weingarz, R. Bergmann, Enhancing Siamese Neural Networks Through Expert Knowledge for Predictive Maintenance, volume 1325 of *CCIS*, Springer, 2020, pp. 77–92.
[61] R. Seiger, M. Franceschetti, B. Weber, An Interactive Method for Detection of Process Activity Executions from IoT Data, Future Internet 15 (2023) 77.
[62] J. Grüger, L. Malburg, R. Bergmann, IoT-enriched event log generation and quality analytics: a case study, it - Information Technology (2023).
[63] R. Seiger, F. Zerbato, A. Burattin, L. Garcia-Banuelos, B. Weber, Towards IoT-driven Process Event Log Generation for Conformance Checking in Smart Factories, in: 24th EDOC Workshops, IEEE, 2020, pp. 20–26.
[64] M. M. Richter, Knowledge Containers, in: Readings in CBR, MKP, 2003.
[65] C. Shearer, The CRISP-DM Model: The New Blueprint for Data Mining, J. Data Warehous 5 (2000) 13–22.
[66] P. Klein, L. Malburg, R. Bergmann, FTOnto: A Domain Ontology for a Fischertechnik Simulation Production Factory by Reusing Existing Ontologies, in: LWDA 2019 Workshop Proc., volume 2454, CEUR-WS.org, 2019, pp. 253–264.
[67] J. Mangler, J. Grüger, L. Malburg, M. Ehrendorfer, Y. Bertrand, J.-V. Benzin, S. Rinderle-Ma, E. Serral Asensio, R. Bergmann, DataStream XES Extension: Embedding IoT Sensor Data into Extensible Event Stream Logs, Future Internet 15 (2023).
[68] C. W. Günther, E. Verbeek, XES Standard Definition Version 2.0, Eindhoven University of Technology (2014).
[69] L. Malburg, J. Grüger, R. Bergmann, An IoT-Enriched Event Log for Process Mining in Smart Factories, Zenodo, 2023. doi:10.6084/m9.figshare.20130794.