Integration of Time Series Embedding for Efficient Retrieval in Case-Based Reasoning*

Justin Weich[1,2], Alexander Schultheis[1,2], Maximilian Hoffmann[1,2], and Ralph Bergmann[1,2]

[1] Artificial Intelligence and Intelligent Information Systems, Trier University, 54296 Trier, Germany, http://www.wi2.uni-trier.de
[2] German Research Center for Artificial Intelligence (DFKI) Branch Trier University, Behringstraße 21, 54296 Trier, Germany
{s4juweic,schultheis,hoffmannm,bergmann}@uni-trier.de
{Justin.Weich,Alexander.Schultheis,Maximilian.Hoffmann,Ralph.Bergmann}@dfki.de

Abstract The increasing volume of time series data in Industry 4.0 applications creates substantial challenges for real-time data analysis. Such analyses that are conducted in the research area of *Temporal Case-Based Reasoning* (TCBR) face performance problems due to complex similarity measures. One potential approach already proven in other domains for addressing these problems is the usage of embedding techniques for time series data, which map these data into a simplified vector representation. Therefore, this paper investigates the integration of time series embedding techniques in the context of *Case-Based Reasoning* (CBR) to improve retrieval efficiency. Therefore, requirements for the application of embedding techniques in CBR are derived. A systematic literature study identifies possible approaches that are analyzed based on the requirements, with the result that no approach is suitable for the application. Therefore, a novel embedding architecture is proposed, using a Siamese neural network approach that can be trained with similarity values. The architecture is prototypically implemented in the ProCAKE framework and evaluated in an Internet of Things use case from a smart factory. The results demonstrate that the embedding-based retrieval achieves classification performance comparable to traditional similarity measures while significantly reducing retrieval time.

Keywords: Temporal Case-Based Reasoning · Time Series Data · Time Series Embedding · Time Series Similarity Measure · Siamese Neural Networks

1 Introduction

Industry 4.0 (I4.0) [18] describes the technological change achieved by the integration of digital technologies into industrial processes. Within the framework

* The final authenticated publication is available online at https://doi.org/10.1007/978-3-031-96559-3_22

of the *Internet of Things* (IoT) - a network of interconnected devices that collect and exchange data -, numerous components in manufacturing environments are equipped with sensors that continuously generate time series data, providing real-time insights into operational conditions. The sensor data generated during industrial manufacturing processes serves as a valuable resource for analysis and control purposes [50]. Analyzing time series data enables the identification of patterns and anomalies, which is crucial for understanding and optimizing processes. The practical applications for this cover various use cases. For instance, in the context of *Event and Activity Detection* (EaAD) [39, 54], sensor data is leveraged to determine ongoing activities within production environments, providing insights into operational processes. Similarly, *Data Quality Issue* (DQI) management [11,32] focuses on identifying inconsistencies in recorded data to ensure reliable analyses and informed decision-making, minimizing the risks caused by erroneous data. Moreover, analyzing time series data can facilitate the early detection of potential failures, as demonstrated in *Predictive Maintenance* (PredM) [49, 72], enabling proactive interventions and minimizing downtime. While *Machine Learning* (ML) methods are often used to analyze such time series data [15, 17], approaches from the research field of *Case-Based Reasoning* (CBR) [1, 9] also offer a feasible *Artificial Intelligence* (AI) methodology for these tasks [39, 52]. This is being investigated in the sub-research area of *Temporal Case-Based Reasoning* (TCBR) [29, 38]. This approach has the advantages that the case returned as a solution provides an initial explanation for the result [58] and that domain knowledge is easier to integrate [66].

The quality of the results of a CBR approach depends on the underlying similarity of the retrieval. In TCBR, established measures are utilized to assess the similarity between the time series of a query and that of a case [39, 48], e.g., the *Smith-Waterman-Algorithm* (SWA) [57] or *Dynamic Time Warping* (DTW) [47]. However, such traditional time series similarity measures often are affected by quadratic computational complexity [48], which becomes particularly problematic with long time series and large case bases. Especially when used in large industrial facilities, where numerous sensors produce different time series, this creates a computational burden which creates substantial barriers to real-time data processing and analysis, especially in time-sensitive industrial applications requiring rapid insights and near real-time results.

Various methods are already considered in CBR for reducing the complexity of time series data by simplifying and shortening these, such as abstraction [56]. However, a drawback of such methods is that patterns or dependencies in the data might get lost through the process. To mitigate this issue, alternative approaches are required. Such a solution is the application of embedding methods, which leverage ML techniques to retain relevant structures while transforming the data into a more compact representation. By mapping time series features into a vector space, embeddings can substantially reduce dimensionality while preserving critical structural information [6, 25]. Specifically, embedding models based on *Neural Networks* (NNs) offer the capability to recognize complex data patterns and represent them efficiently. Therefore, this paper examines the use

of embeddings in TCBR. To this end, a literature study of existing embedding approaches is presented. The suitability of these approaches is to be examined based on the requirements collected, whereupon an embedding-based similarity calculation is designed for the retrieval step in CBR.

The further structure of the paper is as follows: In Sect. 2, the theoretical foundations and related work are presented. Based on the application of embeddings in CBR, requirements are derived in Sect. 3. Sect. 4 contains the methodology and the results of the literature review on available embedding methods for time series, as well as their evaluation, based on the requirements. Building upon the results, Sect. 5 presents the architecture for the time series embedding approach and its integration into the CBR cycle. The evaluation of this approach using a prototype implementation is described in Sect. 6. Subsequently, a conclusion is drawn in Sect. 7 and future work is proposed.

2 Foundations and Related Work

This section provides the necessary background for examining the use of embeddings in TCBR, as outlined in this paper. Therefore, the research field of TCBR is introduced in Sect. 2.1. Sect. 2.2 presents embedding methods in the general context of ML, while Sect. 2.3 reviews related work on embedding of time series and similar structures in general, as well as in CBR.

2.1 Temporal Case-Based Reasoning

TCBR is a sub-area of CBR that deals with the processing and analysis of time series [29, 38]. The focus is on the expression of temporal relationships in cases, with the time series being the most common representation form [39]. A time series depicts real measured values over a period of time, for example, IoT sensor data from manufacturing processes [39, 52]. To assess similarity values between such time series, various measures are developed [39]. Two important algorithms are SWA [57] and DTW [47]. SWA enables the comparison of sequences of different lengths by insertion and deletion operations, while DTW can also consider expansions and compressions in the time series [48]. For the overall similarity calculation, the local-global principle [9, pp. 106–107] is applied. First, local similarities of individual time points are calculated and then aggregated on time series level (global) using the mentioned measures. Afterward, these are again combined on a higher global level, e.g., with similarity values of other time series or global attributes regarding the entire process. The runtime of this similarity calculation is significantly influenced by the similarity measures used for the time series. This is because for each local similarity calculation between the time series elements of the query, there are various options for mapping this to a suitable equivalent of the case. Therefore, a search problem results in which the alignment must be found that maximizes the similarity at the global time series level. Due to this, e.g., SWA and DTW require a quadratic runtime [48], leading to runtime problems, especially with long time series.

2.2 Neural Networks and Embeddings

As a sub-field of AI, ML deals with the development of algorithms that can learn from data and adapt to specific use cases [42]. This is achieved via models that are taught and optimized using training data [4,12]. The prominent research field of deep learning investigates NNs that serve as a powerful ML method. NNs consist of several layers, with each layer containing multiple nodes (called neurons). The fundamental structure comprises an input layer, any number of hidden layers and an output layer. There are weighted connections between the neurons of neighboring layers that enable signal transmission. The weighting of these connections is adjusted during the training process to optimize the prediction accuracy of the model. This training process can be carried out using, e.g., supervised learning [4]. There, the model is presented with the correct solution, allowing the parameters to be adjusted accordingly.

Such NNs can be used to learn embeddings. This is a way of representing complex data in low-dimensional vectors [14, 25]. This is achieved by a mapping function $f : X \to \mathbb{R}^d$ that transforms objects into a d-dimensional vector space. Embeddings are used in various ML areas, such as Large Language Models [70], but can also be used on time series data. An NN is trained for transforming data into embeddings, which learns the properties of the data to be reflected in the vector representation. This transformation enables efficient similarity calculations, as these can be carried out at the vector level. Various measures such as the Euclidean distance or a cosine similarity measure can be implemented to calculate similarities and distances, respectively, between embeddings [35]. For the application of these measures to the embedding NN, a Siamese architecture is used [25, 71]. This is characterized by using two parallel NNs with identical architecture and weights for embedding generation and calculating the similarity value between these mappings. Based on this output, the weights in the NN can be adjusted during training. The application of this embedding approach usually enables both, the reduction of the dimensionality of complex data structures and the preservation of their essential characteristics.

2.3 Related Work

The application of embedding methods in the context of CBR covers various research areas, in particular the processing of time series, their temporal abstraction, and the embedding of related data structures.

Malburg et al. [39] identify four major use cases of CBR for time series data, including prediction, classification, error detection, and application in the medical field. For each of these areas, there is plenty of research outside CBR that deals with time series embeddings to solve the specific use cases mentioned. Lim and Zhoren [37] provide an overview of time series forecasting methods and identify weaknesses of those using embeddings, such as limited adaptability to different time intervals and high data complexity. The applications range from financial to energy markets, where complex influences such as weather or seasonal patterns can be integrated [46, 55]. Time series embeddings also

enable efficient classifications by projecting them into a uniform vector space [23]. Smaller distances between vectors correlate with a higher similarity of the time series. Embeddings also facilitate error detection by making long-term patterns and subtle deviations visible [30, 36].

Temporal abstraction is an alternative approach in which time series are transformed by semantic or symbolic simplification. This includes methods such as temporal abstraction according to Shahar [56] and Allen intervals [2], which define basic relationships between time intervals. Höppner [27] presents further abstraction methods, including time series classification using segmentation and deductive segmentation. These approaches can have the disadvantage that information about patterns or dependencies is lost because, in contrast to embedding, the data representation is not learned but transferred, e.g., on a rule base, into the abstract form.

In the context of related data structures, sequence and graph embeddings are particularly relevant [48]. Sequence embeddings are widely used in language processing and bioinformatics [61], for example in the analysis of protein structures [7]. In the area of graph embeddings, Hoffmann et al. [25] present an innovative approach using graph NNs in a Siamese architecture, which transforms the graph structure into embeddings and calculates similarity values between those. Other application areas include knowledge graphs and recommendation systems [63], where graph embeddings are used for complexity reduction and efficient operationalization. Klein et al. [34] demonstrate the application of embeddings for workflow optimization in CBR using knowledge graph embeddings.

3 Requirements for Using Time Series Embedding in Case-Based Reasoning

To select suitable embedding methods for time series, requirements for these methods are needed. Some come from the general application scenario in CBR, others result from the application of ML models. The following four requirements are identified.

Req. 1 Applicability to Time Series: The embedding method must either be directly applicable to time series or embed sequences or graphs, since time series can be represented as these [48]. The requirement for graph or sequence embedders is that they take the temporal component of the data into account. The embedding method must use (encoded) time series (or sequences or graphs) with variable time components as input. The specification is therefore ideally a direct time series embedding.

Req. 2 Data Types: The embedding procedure must be applicable to time series with different data types used for the entries, i.e., the timestamp itself as well as the measured data value. The requirement stems from the CBR use case, as complex data objects are contained in the time series. For example, it must be possible to embed time series with numerical data or with entire objects. The process should be able to automatically recognize and process the data types.

Req. 3 Trainability Based on Similarity Values: It must be possible to train the embedding process using similarity values as input parameters. The similarity values are provided as input so that the method can train an ML model using these values. This is necessary to automatically create mapping functions for suitable embeddings. By using the similarity values, this can be done independently of the underlying similarity measures so that the embedding approach can be used for different domains and use cases. This is a form of supervised learning (see Sect. 2.2), as the model is trained as closely as possible to the given similarity values.

Req. 4 Embedding Dimension: The embedding procedure should have a manually adjustable embedding dimension or automatically determine the optimum dimension. This ensures that an optimal embedding of the time series is generated and the quality of the embeddings is improved.

4 Literature Analysis on Time Series Embedding

To identify the current state of the research in the field of embedding methods for time series, a systematic literature review [45] was carried out[1] on the platforms Google Scholar[2], Semantic Scholar[3], Scopus[4] and IEEE Xplore[5]. The sources found were further analyzed using forward and backward snowballing. The review was not only conducted for time series embedding methods, but also for sequence and graph embedding methods, as time series can be represented as sequences or graphs (see Req. 1).

A total of ten methods for embedding time series data were identified, which are illustrated in Tab. 1. An evaluation based on the requirements derived is also included there. Furthermore, the maturity of the approaches was also considered by checking whether an open-source implementation is available. A result of the requirements analysis shows that Req. 3 is not fulfilled by any of the embedding methods. This is because the methods already use built-in similarity measures and cannot be trained using similarity values. The disadvantage is that the methods are adapted to a specific similarity measure, usually DTW, and are therefore not generically applicable without adaptation. Methods that can be used with similarity values as input parameters, such as Multidimensional Scaling according to Torgerson [60], do not include a ML model and therefore do not provide a training (see Req. 3). Irrespective of Req. 3, only TS2Vec [62] as a pure time series embedding fulfills the other requirements at all. The methods for sequences and graphs show similar restrictions. Therefore, the requirements' analysis comes to the conclusion that none of the existing methods is suitable for the application of time series embedding in CBR.

[1] The literature review was carried out in May 2024.
[2] https://scholar.google.de/
[3] https://www.semanticscholar.org/
[4] https://www.scopus.com/
[5] https://ieeexplore.ieee.org/

Table 1. Comparison of the Requirements of the Ten Identified Embedding Approaches. (✓ = Suitable, ✗ = Unsuitable, (✓) = Partially Suitable.)

Embedding Approach	Req. 1	Req. 2	Req. 3	Req. 4	Implementation Available?
Boosted Embedding [31]	✓	✗	✗	✗	✗
Embed-RUL [21]	✓	✗	✗	✗	✗
Embedding in Pseudo-Euclidean Space by Graepel et al. [20]	✓	✓	✗	✓	✗
Laplacian Eigenmap by Belkin and Niyogi [5]	✓	✓	✗	✓	✗
LSTM Embedding by Uribarri and Mindlin [62]	✓	✓	✗	✓	✗
Multidimensional Scaling by Torgerson [60]	✓	✓	✗	✓	(✓)[6]
Random Warping Series [68]	✓	✗	✗	✓	✓[7]
Signal2vec [43]	✓	✗	✗	✗	✓[8]
Time2Vec [41]	✓	✗	✗	✓	✓[9]
TS2Vec [62]	✓	✓	✗	✓	✓[10]

In addition to the investigated embedding methods for time series data, further problems could be recognized in the identified methods for sequences and graphs. Overviews of sequence embedding methods are presented for application areas as, e.g., biological sequence embeddings by Tran et al. [61] or Ibtehaz and Kihara [28], as well as for word embeddings by Wang et al. [65]. Sequence embeddings generally do not support consideration of the time point, but only of the sequentially recorded value. This means that information would be lost during embedding if the measured values are not collected at homogeneous time intervals. The loss of this relevant information, which is considered in traditional similarity calculations, contradicts Req. 2. Despite this exclusion argument, several methods were investigated, e.g., Autoencoder & seq2seq by Guo et al. [22], and seq2vec [33]. This analysis is available in an additional document[11]. In supplement to the problem already described with Req. 2, e.g., the sequence embeddings considered also do not fulfill Req. 3, as they are often not based on similarities, and also have other difficulties such as the time series embeddings. Further, graph embeddings were also considered, for which various overviews are available, too, such as Cai et al. [13], Goyal and Ferrara [19], Xu [69], Wang et

[6] Partially available: https://github.com/drewwilimitis/Manifold-Learning
[7] https://github.com/IBM/RandomWarpingSeries
[8] https://github.com/ChristoferNal/multi-nilm
[9] https://github.com/ojus1/Time2Vec-PyTorch
[10] https://github.com/zhihanyue/ts2vec
[11] Complete requirements' analysis available at: https://gitlab.rlp.net/procake-embedding/procake-embedding-time-series-evaluation/-/blob/main/Requirements-Analysis.pdf

al. [64], and Barros et al. [3]. When transforming time series data into graphs, the problem arises that there is too much irrelevant information in the more extensive graph structure. For example, the information of the time points with associated values would be available as nodes, whereas the edges have no semantic meaning. As a result, a lot of irrelevant information is considered during embedding, which leads to an unnecessary increase in complexity and therefore runtime. Various methods, e.g., graph2Vec [44], and, StarSpace [67], were considered in the evaluation and the same difficulties arose as with the other embedding methods. For example, no method was found that fulfilled Req. 3. Due to the expected difficulties caused by the conversion to graph format and the resulting irrelevant information, no further analysis of the graph methods was carried out on this basis. The analysis of the considered embedding methods for graphs is also included in the additional document[11].

5 Application of Time Series Embedding in Similarity-Based Retrieval

As explained in Sect. 3, none of the existing embedding approaches fulfill the requirements for being used in CBR. Therefore, a generic embedding approach for time series that meets the requirements is designed in Sect. 5.1. Its integration into the CBR cycle is presented in Sect. 5.2.

5.1 Architecture of the Time Series Embedding Approach

The developed architecture for a time series embedding applicable in CBR is visualized in Fig. 1. It is built on a layer-based approach that uses established methods for NN procedures in IoT environments. The system processes two time series and calculates their similarity using three specialized layers:

(1) **Preprocessing Layer**: This layer is responsible for processing the time series data, which includes encoding and normalization. First, the processing of different data types is enabled by transforming these into numerical representations (see Req. 2). This encoding approach can be adapted from Hoffmann et al. [25], who develop encodings for different data types in CBR, originally, for graph embeddings. Their method first encodes atomic elements with type-specific methods, then builds a tree-encoding in a bottom-up fashion to preserve the hierarchical structure for NN processing. Therefore, the encoding step provides a foundation for generic encoding while maintaining compatibility with case representations used in a CBR application. Second, the encoded time series are normalized by conversion into vector sequences, which are necessary for *Long Short-Term Memory* (LSTM) processing. It extracts and concatenates vector information from tree nodes, enriching it with implicit structural data. During training, sequences are normalized to match the longest sequence in the training dataset through padding.

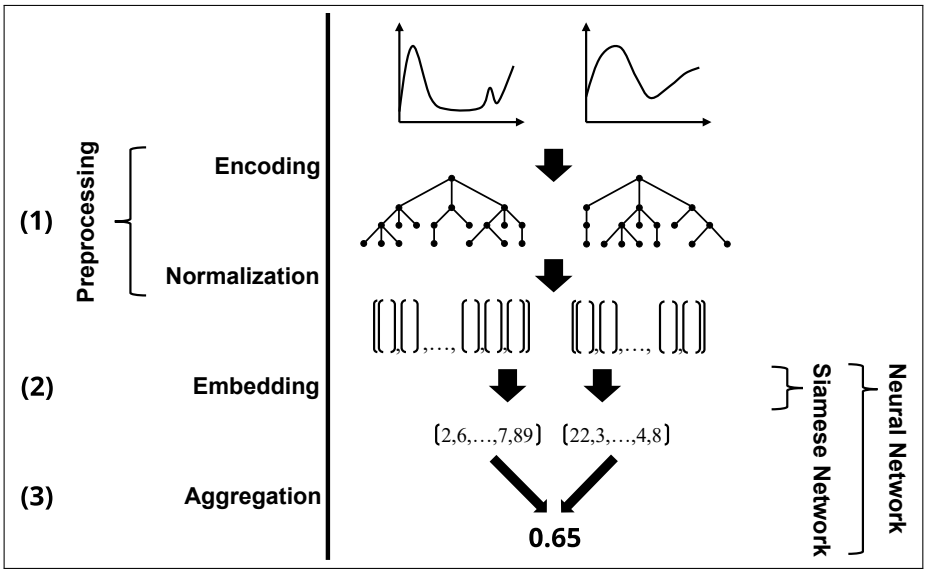

Fig. 1. The Architecture of the Time Series Embedding Approach.

(2) **Embedding Layer**: In this layer, the normalized vector sequences are processed by multiple LSTM layers of a Siamese network with an adjustable embedding dimension (see Req. 3). The first LSTM layer outputs a complete sequence, while the second LSTM layer produces a condensed state vector, followed by two multi layer perceptrons with *Rectified Linear Unit* (ReLU) activation. This architecture enables the network to learn complex dependencies and generate embeddings for both input time series.
If an embedding has already been calculated for a time series, for example for a query in a retrieval, the embedding is cached, and these two layers (preprocessing, and embedding) are skipped.

(3) **Aggregation Layer**: Finally, this layer combines the generated embeddings and calculates their similarity using a single-neuron dense layer with sigmoid activation. This layer directly outputs similarity values in the interval $[0, 1]$, eliminating the need for additional normalization steps. The design choice of computing similarity rather than returning embeddings optimizes for CBR system requirements while maintaining model simplicity.

The architecture also enables the processing of variable time series lengths through appropriate normalization and masking during training.

5.2 Integration in the Case-Based Reasoning Cycle

The developed architecture is integrated into a CBR system by adding the embedding approach presented in Sect. 5.1 to the knowledge container of the similarity measures. The underlying NN is trained once based on the case base and

similarity values calculated between the cases with a traditional measure, e.g., SWA or DTW. The call of the embedding system is made when a similarity calculation is pending for a time series, which can be nested in a higher-level object (cf. local-global principle, Sect. 2.1). The time series contained in query and case are encoded and normalized, followed by embedding and the similarity calculation (corresponding to the layers in Sect. 5.1). Already determined embeddings can be stored in the case base to increase efficiency. The computed similarity value as output of the NN is transmitted back to the CBR application for further processing in the retrieval step.

6 Experimental Evaluation

To evaluate the embedding approach, it is compared with the traditional DTW method. This takes place for the EaAD use case, for which Malburg et al. [39] already present a successful case-based approach using DTW. There, CBR is used to classify which event or activity is being executed for the time series data contained in the query. In the context of this comparison, the following hypotheses are investigated:

Hypothesis 1 (H1). The usage of embedding-based similarity measures accelerates retrieval compared to the usage of DTW.

Hypothesis 2 (H2). The embedding-based CBR approaches delivers results comparable in quality to the DTW-based approach.

- **H2a.** The embedding-based CBR approach delivers retrieval results which ranking order is correct and complete compared to the DTW-based.
- **H2b.** The embedding-based CBR approach achieves classification performance comparable to the DTW-based.

Before the examination of these hypotheses, the implementation of the approach as well as the setup of the evaluation is described in Sect. 6.1. Then, the comparison results are presented in Sect. 6.2 and the answers to the hypotheses are discussed.

6.1 Implementation and Experimental Setup

The presented embedding approach is implemented in the open-source CBR framework ProCAKE[12] [8,10], which is written in Java [53]. The EaAD use case is already available there, for which implementation details are presented by Malburg et al. [39]. The encoding layer is also integrated in ProCAKE and sends its output via an interface to the other layers, which are realized in Python using the TensorFlow libraries [59]. This implementation is encapsulated in a Docker

[12]Implementation available at: https://gitlab.rlp.net/procake-embedding/procake-embedding-core/

image and returns the resulting similarity value to the ProCAKE framework. To increase efficiency, the similarity calculation in the NN is carried out in batches.

For the evaluation[13], we use data from a Fischertechnik Smart Factory located in the IoT Lab Trier[14] [40] that was evaluated for EaAD before [39]. A case base of 1,100 cases is used here, containing unprocessed time series data. Of these cases, the time series from 1,000 cases were used for training. For this purpose, similarity values were calculated between all these time series using DTW (a case contains ten different time series in this domain) and the embedding system was trained based on these values. The training took place on a `Tesla V100-SXM2-32GB` graphics card with `Intel(R) Xeon(R) Gold 6138 CPU @ 2.00GHz` processor. Each epoch required approximately 50 hours and 30 minutes, resulting in a total training time of 101 hours for two epochs with a batch size of 128, excluding data import. In total, 106,565 parameters were initialized, of which 35,521 are trainable and 71,044 are optimizer parameters. The model expects input vectors of arbitrary length and a dimensionality of 19, based on the encoding scheme proposed by Hoffmann et al. [25]. During inference, each input is transformed into an embedding of length 16, which is then compared to a reference embedding in the aggregation layer to compute a similarity value. Using this embedding approach as time series similarity measure, a retrieval was performed with the remaining 100 cases not given to the training as queries. For comparison, a baseline retrieval was also performed using the traditional DTW approach.

6.2 Experimental Results and Discussion

The evaluation has been carried out on a server with 34 processors, a clock frequency of 2,850 MHz, and 400 gigabytes of RAM. The embedding-based and the DTW-based CBR approach were executed on this setup for the EaAD use case. To answer H1, the times required for the retrieval execution are compared. In both cases, this did not run in parallel, but on one thread. For the embedding-based CBR approach, the retrieval required a runtime of 8 minutes and 43 seconds. This includes 1 minute and 46 seconds spent on encoding and embedding the case base, which could also be omitted by permanent storage. In contrast, the DTW-based approach required a runtime of 3 hours and 47 minutes. These results indicate that a runtime acceleration is achieved by using the embedding methods, so that H1 can be confirmed.

To answer H2, the ten most similar cases returned from the retrieval are considered. For the investigation of H2a, correctness and completeness are used as ranking measures for the lists of retrieval results [26]. Correctness provides information about the same order of the results, whereas completeness indicates whether both lists contain the same elements. The average correctness is 0.055 and the median 0.0, which is in the middle of the value range $[1.0, -1.0]$. These

[13]Runnable evaluation available at: https://gitlab.rlp.net/procake-embedding/procake-embedding-time-series-evaluation

[14]https://iot.uni-trier.de/

values indicate that the correctness is unfavorable, as this value implies an almost random distribution of the ranked cases. The completeness is 0.094 on average and 0.0 in the median, with the latter representing the minimum value. This suggests that H2a can be rejected for the use case of EaAD.

Table 2. Performance Measures Calculated Based on the Classifications of the Embedding-Based and DTW-Based CBR Approaches.

	Accuracy	**Precision**	**Recall**	**F1-Score**	**Specificity**	**False Positive Rate**	**False Negative Rate**
DTW-based Approach	0.92	0.8	0.8	0.8	0.95	0.05	0.2
Embedding-based Approach	0.848	0.62	0.62	0.62	0.905	0.095	0.38
Δ between Approaches	0.072	0.18	0.18	0.18	0.045	0.045	0.18

To consider H2b, the queries are classified in the adaptation step. This is again based on the ten most similar cases returned from the retrieval phase and carried out using majority voting. For the DTW-based CBR approach, 80 of the 100 queries are classified correctly and 20 incorrectly, whereby for the embedding-based approach, 62 queries correctly and 20 incorrectly. Therefore, 18 classifications differ between the approaches. The performance measures for the methods are calculated based on a confusion matrix, which is illustrated in Tab. 2. Precision, recall, the F1 score and the false negative rate are relatively far apart with a $\Delta = 0.18$, whereas the accuracy, specificity and false negative rate are close to each other. H2b can therefore also tend to be rejected, whereby the entire hypothesis H2 must be rejected.

The results presented for H2 indicate that the embedding-based approach performs qualitatively worse than the DTW-based. However, H1 implies that the embedding approach is significantly faster, so that a trade-off between these two parameters can be assumed. The measures of completeness and correctness considered in H2a provide only limited information about quality in this context. As a reason for this, it was found in a detailed inspection that for some queries there are several cases whose similarity values are very close to each other (in some cases the most similar cases differ by $\Delta < 0.01$). This means that when only the ten most similar cases are considered, these small differences have a large impact on these metrics, resulting in poor values for correctness and completeness. The analysis of H2b shows that the difference in performance is not that much substantial, but still considerable.

7 Conclusion and Future Work

This paper investigates the integration of time series embeddings as a similarity measure in CBR. After a systematic literature analysis of existing embedding methods, requirements for the use of such methods in CBR are identified. This

analysis indicates that none of the existing approaches meets the defined requirements. A new embedding method for time series is then designed considering the requirements. For the evaluation, this approach is implemented as a proof-of-concept in the CBR framework ProCAKE. The evaluation shows that the runtime of the retrieval using embedding could be significantly reduced, but partly at the expense of the quality of results.

Based on the results of this work, a further optimization of the embedding approach can be investigated, e.g., by research on hyperparameter optimization [24]. Furthermore, for the EaAD use case, it can be examined whether better performance can be achieved by autoencoders, which learn independently of DTW and possibly outperform it as the previously established CBR approach. Semantic knowledge can also be incorporated into similarity calculations to ensure that the analysis goes beyond mere syntactic information. In addition to this investigated use case, embeddings can also be used for other computationally intensive domains in CBR such as DQIs or PredM. In the DQI context, it can also be investigated how embeddings learn a similarity assignment model from manually defined similarities, or how the use of autoencoders can be further researched. Also, the use of NN reduces the understandability of the similarity calculation. In future research, approaches to explain the similarity calculation can be investigated, as already exists for other data structures (e.g., [51]). In addition, embedding in CBR can also be investigated for MAC/FAC approaches [16] as a pre-filtering method in the MAC phase.

Acknowledgments. This work is funded by the Federal Ministry for Economic Affairs and Climate Action under grant No. 01MD22002C *EASY* [50].

References

1. Aamodt, A., Plaza, E.: Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Commun. **7**(1), 39–59 (1994). https://doi.org/10.3233/AIC-1994-7104
2. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Commun. ACM **26**(11), 832–843 (1983). https://doi.org/10.1145/182.358434
3. de Barros, C.D.T., Mendonça, M.R.F., Vieira, A.B., Ziviani, A.: A Survey on Embedding Dynamic Graphs. ACM Comput. Surv. **55**(2), 10:1–10:37 (2023). https://doi.org/10.1145/3483595
4. Başţanlar, Y., Özuysal, M.: Introduction to Machine Learning, pp. 105–128. Humana Press, Totowa, NJ (2014). https://doi.org/10.1007/978-1-62703-748-8_7
5. Belkin, M., Niyogi, P.: Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering. In: NIPS 2001 Proc. pp. 585–591. MIT Press (2001). https://doi.org/10.5555/2980539.2980616
6. Bengio, Y., Courville, A.C., Vincent, P.: Representation Learning: A Review and New Perspectives. IEEE Trans. Pattern Anal. Mach. Intell. **35**(8), 1798–1828 (2013). https://doi.org/10.1109/TPAMI.2013.50
7. Bepler, T., Berger, B.: Learning protein sequence embeddings using information from structure. In: 7th ICLR Proc. OpenReview.net (2019). https://doi.org/10.48550/arXiv.1902.08661

8. Bergmann, R., Grumbach, L., Hoffmann, M., Jung, L., Malburg, L., Schultheis, A., Zeyen, C.: ProCAKE Framework (7.1.0) (2025). https://doi.org/10.5281/zenodo.15222500
9. Bergmann, R.: Experience Management: Foundations, Development Methodology, and Internet-Based Applications, LNCS, vol. 2432. Springer (2003). https://doi.org/10.1007/3-540-45759-3
10. Bergmann, R., Grumbach, L., Malburg, L., Zeyen, C.: ProCAKE: A Process-Oriented Case-Based Reasoning Framework. In: 27th ICCBR Workshop Proc. vol. 2567, pp. 156–161. CEUR-WS.org (2019)
11. Bertrand, Y., Schultheis, A., Malburg, L., Grüger, J., Serral Asensio, E., Bergmann, R.: Challenges in Data Quality Management for IoT-Enhanced Event Logs. In: 19th RCIS Proc. pp. 20–36. LNBIP, Springer (2025). https://doi.org/10.1007/978-3-031-92474-3_2
12. Bishop, C.M.: Pattern recognition and machine learning, 5th Edition. Information science and statistics, Springer (2007)
13. Cai, H., Zheng, V.W., Chang, K.C.: A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications. IEEE Trans. Knowl. Data Eng. **30**(9), 1616–1637 (2018). https://doi.org/10.1109/TKDE.2018.2807452
14. Cellucci, C.J., Albano, A.M., Rapp, P.E.: Comparative study of embedding methods. Phys. Rev. E **67**, 066210 (2003). https://doi.org/10.1103/PhysRevE.67.066210
15. Choi, K., Yi, J., Park, C., Yoon, S.: Deep Learning for Anomaly Detection in Time-Series Data: Review, Analysis, and Guidelines. IEEE Access **9**, 120043–120065 (2021). https://doi.org/10.1109/ACCESS.2021.3107975
16. Forbus, K.D., Gentner, D., Law, K.: MAC/FAC: A Model of Similarity-Based Retrieval. Cogn. Sci. **19**(2), 141–205 (1995). https://doi.org/10.1207/s15516709cog1902_1
17. Gamboa, J.C.B.: Deep Learning for Time-Series Analysis. CoRR **abs/1701.01887** (2017). https://doi.org/10.48550/arXiv.1701.01887
18. Gilchrist, A.: Industry 4.0: The Industrial Internet of Things. Apress (2016). https://doi.org/10.1007/978-1-4842-2047-4
19. Goyal, P., Ferrara, E.: Graph embedding techniques, applications, and performance: A survey. Knowl. Based Syst. **151**, 78–94 (2018). https://doi.org/10.1016/J.KNOSYS.2018.03.022
20. Graepel, T., Herbrich, R., Bollmann-Sdorra, P., Obermayer, K.: Classification on Pairwise Proximity Data. In: NIPS 1998 Proc. pp. 438–444. The MIT Press (1998). https://doi.org/10.5555/3009055.3009117
21. Gugulothu, N., TV, V., Malhotra, P., Vig, L., Agarwal, P., Shroff, G.: Predicting Remaining Useful Life using Time Series Embeddings based on Recurrent Neural Networks. IJPHM **9**(1) (2018). https://doi.org/10.48550/arXiv.1709.01073
22. Guo, R., Fujiwara, T., Li, Y., Lima, K.M., Sen, S., Tran, N.K., Ma, K.: Comparative visual analytics for assessing medical records with sequence embedding. Vis. Informatics **4**(2), 72–85 (2020). https://doi.org/10.1016/J.VISINF.2020.04.001
23. Hayashi, A., Mizuhara, Y., Suematsu, N.: Embedding Time Series Data for Classification. In: 4th MLDM Proc. LNCS, vol. 3587, pp. 356–365. Springer (2005). https://doi.org/10.1007/11510888_35
24. Hoffmann, M., Bergmann, R.: Improving Automated Hyperparameter Optimization with Case-Based Reasoning. In: 30th ICCBR Proc. LNCS, vol. 13405, pp. 273–288. Springer (2022). https://doi.org/10.1007/978-3-031-14923-8_18

25. Hoffmann, M., Bergmann, R.: Using Graph Embedding Techniques in Process-Oriented Case-Based Reasoning. Algorithms **15**(2), 27 (2022). https://doi.org/10.3390/A15020027
26. Hoffmann, M., Malburg, L., Klein, P., Bergmann, R.: Using Siamese Graph Neural Networks for Similarity-Based Retrieval in Process-Oriented Case-Based Reasoning. In: 28th ICCBR Proc. LNCS, vol. 12311, pp. 229–244. Springer (2020). https://doi.org/10.1007/978-3-030-58342-2_15
27. Höppner, F.: Time Series Abstraction Methods - A Survey. In: INFORMATIK 2002 Proc. LNI, vol. P-19, pp. 777–786. GI (2002)
28. Ibtehaz, N., Kihara, D.: Application of Sequence Embedding in Protein Sequence-Based Predictions, pp. 31–55. World Scientific (2023). https://doi.org/10.1142/9789811258589_0002
29. Jære, M.D., Aamodt, A., Skalle, P.: Representing Temporal Knowledge for Case-Based Prediction. In: 6th ECCBR Proc. LNCS, vol. 2416, pp. 174–188. Springer (2002). https://doi.org/10.1007/3-540-46119-1_14
30. Ji, Z., Wang, Y., Yan, K., Xie, X., Xiang, Y., Huang, J.: A space-embedding strategy for anomaly detection in multivariate time series. Expert Syst. Appl. **206**, 117892 (2022). https://doi.org/10.1016/J.ESWA.2022.117892
31. Karingula, S.R., Ramanan, N., Tahmasbi, R., Amjadi, M., Jung, D., Si, R., Thimmisetty, C., Polanía, L.F., Sayer, M., Taylor, J., Coelho, C.N.: Boosted Embeddings for Time-Series Forecasting. In: 7th LOD Proc. LNCS, vol. 13164, pp. 1–14. Springer (2021). https://doi.org/10.1007/978-3-030-95470-3_1
32. Karkouch, A., Mousannif, H., Al Moatassime, H., Noel, T.: Data Quality in Internet of Things: A state-of-the-art survey. JNCA **73**, 57–81 (2016). https://doi.org/10.1016/j.jnca.2016.08.002
33. Kim, H.J., Hong, S., Cha, K.J.: seq2vec: Analyzing sequential data using multi-rank embedding vectors. Electron. Commer. Res. Appl. **43**, 101003 (2020). https://doi.org/10.1016/J.ELERAP.2020.101003
34. Klein, P., Malburg, L., Bergmann, R.: Learning Workflow Embeddings to Improve the Performance of Similarity-Based Retrieval for Process-Oriented Case-Based Reasoning. In: 27th ICCBR Proc. LNCS, vol. 11680, pp. 188–203. Springer (2019). https://doi.org/10.1007/978-3-030-29249-2_13
35. Kljun, M., Tersek, M.: A review and comparison of time series similarity measures. In: 29th ERK Proc. pp. 21–22 (2020)
36. Li, Z., Zhao, Y., Han, J., Su, Y., Jiao, R., Wen, X., Pei, D.: Multivariate Time Series Anomaly Detection and Interpretation using Hierarchical Inter-Metric and Temporal Embedding. In: 27th SIGKDD Proc. pp. 3220–3230. ACM (2021). https://doi.org/10.1145/3447548.3467075
37. Lim, B., Zohren, S.: Time Series Forecasting With Deep Learning: A Survey. Philos. Trans. R. Soc. A **379**(2194), 20200209 (2021). https://doi.org/10.1098/rsta.2020.0209
38. López, B.: Case-Based Reasoning: A Concise Introduction. SLAIML, Springer (2013). https://doi.org/10.1007/978-3-031-01562-5
39. Malburg, L., Schultheis, A., Bergmann, R.: Modeling and Using Complex IoT Time Series Data in Case-Based Reasoning: From Application Scenarios to Implementations. In: 31st ICCBR Workshop Proc. vol. 3438, pp. 81–96 (2023)
40. Malburg, L., Seiger, R., Bergmann, R., Weber, B.: Using Physical Factory Simulation Models for Business Process Management Research. In: BPM 2020 Workshop Proc. LNBIP, vol. 397, pp. 95–107. Springer (2020). https://doi.org/10.1007/978-3-030-66498-5_8

41. Mehran Kazemi, S., Goel, R., Eghbali, S., Ramanan, J., Sahota, J., Thakur, S., Wu, S., Smyth, C., Poupart, P., Brubaker, M.A.: Time2Vec: Learning a Vector Representation of Time. CoRR **abs/1907.05321** (2019). https://doi.org/10.48550/arXiv.1907.05321
42. Michalski, R.S., Carbonell, J.G., Mitchell, T.M.: Machine Learning - An Artificial Intelligence Approach. Symbolic computation, Springer (1983). https://doi.org/10.1007/978-3-662-12405-5
43. Nalmpantis, C., Vrakas, D.: Signal2Vec: Time Series Embedding Representation. In: 20th EANN Proc. CCIS, vol. 1000, pp. 80–90. Springer (2019). https://doi.org/10.1007/978-3-030-20257-6_7
44. Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., Jaiswal, S.: graph2vec: Learning Distributed Representations of Graphs. CoRR **abs/1707.05005** (2017). https://doi.org/10.48550/arXiv.1707.05005
45. Ridley, D.: The Literature Review: A Step-by-Step Guide for Students. SAGE (2012)
46. Rosato, A., Altilio, R., Araneo, R., Panella, M.: Embedding of time series for the prediction in photovoltaic power plants. In: 16th EEEIC Proc. pp. 1–4 (2016). https://doi.org/10.1109/EEEIC.2016.7555872
47. Sakoe, H., Chiba, S.: Dynamic Programming Algorithm Optimization for Spoken Word Recognition. IEEE Trans. Aco. Sp. and Sig. Proc. **26**(1), 43–49 (1978). https://doi.org/10.1109/TASSP.1978.1163055
48. Schake, E., Grumbach, L., Bergmann, R.: A Time-Series Similarity Measure for Case-Based Deviation Management to Support Flexible Workflow Execution. In: 28th ICCBR Proc. LNCS, vol. 12311, pp. 33–48. Springer (2020). https://doi.org/10.1007/978-3-030-58342-2_3
49. Schultheis, A.: Exploring a Hybrid Case-Based Reasoning Approach for Time Series Adaptation in Predictive Maintenance. In: 32nd ICCBR Workshop Proc. vol. 3708, pp. 230–235. CEUR-WS.org (2024)
50. Schultheis, A., Alt, B., Bast, S., Guldner, A., Jilg, D., Katic, D., Mundorf, J., Schlagenhauf, T., Weber, S., Bergmann, R., Bergweiler, S., Creutz, L., Dartmann, G., Malburg, L., Naumann, S., Rezapour, M., Ruskowski, M.: EASY: Energy-Efficient Analysis and Control Processes in the Dynamic Edge-Cloud Continuum for Industrial Manufacturing. Künstliche Intelligenz (2024). https://doi.org/10.1007/s13218-024-00868-3
51. Schultheis, A., Hoffmann, M., Malburg, L., Bergmann, R.: Explanation of Similarities in Process-Oriented Case-Based Reasoning by Visualization. In: 31st ICCBR Proc. LNCS, vol. 14141, pp. 53–68. Springer (2023). https://doi.org/10.1007/978-3-031-40177-0_4
52. Schultheis, A., Malburg, L., Grüger, J., Weich, J., Bertrand, Y., Bergmann, R., Serral Asensio, E.: Identifying Missing Sensor Values in IoT Time Series Data: A Weight-Based Extension of Similarity Measures for Smart Manufacturing. In: 32nd ICCBR Proc. LNCS, vol. 14775, pp. 240–257. Springer (2024). https://doi.org/10.1007/978-3-031-63646-2_16
53. Schultheis, A., Zeyen, C., Bergmann, R.: An Overview and Comparison of Case-Based Reasoning Frameworks. In: 31st ICCBR Proc. LNCS, vol. 14141, pp. 327–343. Springer (2023). https://doi.org/10.1007/978-3-031-40177-0_21
54. Seiger, R., Schultheis, A., Bergmann, R.: Case-Based Activity Detection from Segmented Internet of Things Data. In: 33rd ICCBR Proc. LNCS, Springer (2025). https://doi.org/10.1007/978-3-031-96559-3_29, Accepted for Publication.

55. Sezer, O.B., Gudelek, M.U., Özbayoglu, A.M.: Financial time series forecasting with deep learning : A systematic literature review: 2005-2019. Appl. Soft Comput. **90**, 106181 (2020). https://doi.org/10.1016/J.ASOC.2020.106181
56. Shahar, Y.: A Framework for Knowledge-Based Temporal Abstraction. Artif. Intell. **90**(1-2), 79–133 (1997). https://doi.org/10.1016/S0004-3702(96)00025-2
57. Smith, T.F., Waterman, M.S.: Identification of Common Molecular Subsequences. JMB **147**(1), 195–197 (1981)
58. Sørmo, F., Cassens, J., Aamodt, A.: Explanation in Case-Based Reasoning – Perspectives and Goals. Artif. Intell. Rev. **24**(2), 109–143 (2005). https://doi.org/10.1007/s10462-005-4607-7
59. TensorFlow Developers: TensorFlow (2024). https://doi.org/10.5281/zenodo.12726004
60. Torgerson, W.S.: Theory and methods of scaling. J. Wiley, New York (1958), section: 460 p.
61. Tran, C., Khadkikar, S., Porollo, A.: Survey of Protein Sequence Embedding Models. Int. J. Mol. Sci. **24**(4) (2023). https://doi.org/10.3390/ijms24043775
62. Uribarri, G., Mindlin, G.B.: Dynamical time series embeddings in recurrent neural networks. Chaos Solit. Fractals **154**, 111612 (2022). https://doi.org/10.1016/j.chaos.2021.111612
63. Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., Wang, Z.: Knowledge-aware Graph Neural Networks with Label Smoothness Regularization for Recommender Systems. In: 25th SIGKDD Proc. pp. 968–977. ACM (2019). https://doi.org/10.1145/3292500.3330836
64. Wang, Q., Mao, Z., Wang, B., Guo, L.: Knowledge Graph Embedding: A Survey of Approaches and Applications. IEEE Trans. Knowl. Data Eng. **29**(12), 2724–2743 (2017). https://doi.org/10.1109/TKDE.2017.2754499
65. Wang, S., Zhou, W., Jiang, C.: A survey of word embeddings based on deep learning. Computing **102**(3), 717–740 (2020). https://doi.org/10.1007/S00607-019-00768-7
66. Watson, I.D., Marir, F.: Case-based reasoning: A review. Knowl. Eng. Rev. **9**(4), 327–354 (1994). https://doi.org/10.1017/S0269888900007098
67. Wu, L.Y., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: StarSpace: Embed All The Things! In: 32nd AAAI Proc. pp. 5569–5577. AAAI Press (2018). https://doi.org/10.1609/AAAI.V32I1.11996
68. Wu, L., Yen, I.E., Yi, J., Xu, F., Lei, Q., Witbrock, M.: Random Warping Series: A Random Features Method for Time-Series Embedding. In: AISTATS 2018 Proc. PMLR, vol. 84, pp. 793–802. PMLR (2018)
69. Xu, M.: Understanding Graph Embedding Methods and Their Applications. SIAM Rev. **63**(4), 825–853 (2021). https://doi.org/10.1137/20M1386062
70. Zhao, W.X., Zhou, K., Li, J., Tang, T., Wang, X., Hou, Y., Min, Y., Zhang, B., Zhang, J., Dong, Z., Du, Y., Yang, C., Chen, Y., Chen, Z., Jiang, J., Ren, R., Li, Y., Tang, X., Liu, Z., Liu, P., Nie, J., Wen, J.: A Survey of Large Language Models. CoRR **abs/2303.18223** (2023). https://doi.org/10.48550/ARXIV.2303.18223
71. Zheng, W., Yang, L., Genco, R.J., Wactawski-Wende, J., Buck, M., Sun, Y.: SENSE: Siamese neural network for sequence embedding and alignment-free comparison. Bioinform. **35**(11), 1820–1828 (2019). https://doi.org/10.1093/BIOINFORMATICS/BTY887
72. Zonta, T., Da Costa, C.A., da Rosa Righi, R., de Lima, M.J., da Trindade, E.S., Li, G.P.: Predictive maintenance in the Industry 4.0: A systematic literature review. Comput. Ind. Eng. **150**, 106889 (2020). https://doi.org/10.1016/j.cie.2020.106889